Titre: Title:	Suivi de cible basé sur un modèle de superpixels et points clés
Auteur: Author:	François-Xavier Derue
Date:	2016
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Derue, F.-X. (2016). Suivi de cible basé sur un modèle de superpixels et points clés [Master's thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/2119/

UNIVERSITÉ DE MONTRÉAL

SUIVI DE CIBLE BASÉ SUR UN MODÈLE DE SUPERPIXELS ET POINTS CLÉS

FRANÇOIS-XAVIER DERUE
DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INFORMATIQUE)
AVRIL 2016

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

SUIVI DE CIBLE BASÉ SUR UN MODÈLE DE SUPERPIXELS ET POINTS CLÉS

présenté par : DERUE François-Xavier
en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées
a été dûment accepté par le jury d'examen constitué de :

M. PESANT Gilles, Ph. D., président
M. BILODEAU Guillaume-Alexandre, Ph. D., membre et directeur de recherche
M. BERGEVIN Robert, Ph. D., membre et codirecteur de recherche
M. HURTUT Thomas, Ph. D., membre

REMERCIEMENTS

Je tiens à exprimer ma plus sincère gratitude à mon directeur de recherche, Professeur Guillaume-Alexandre Bilodeau, pour son encadrement, sa disponibilité, ses conseils ainsi que le soutien financier qu'il a pu m'obtenir auprès du Fonds de recherche du Québec - Nature et technologies (FRQNT). Je remercie également mon codirecteur, Professeur Robert Bergevin, pour ses révisions détaillées et les commentaires pertinents qu'il m'a apportés tout au long de ce travail. Je salue mes collègues et amis du laboratoire LITIV, Pierre-Luc, Tanushri, Farnoosh et Hui-Lee, pour m'avoir partagé leur expérience et leur savoir. Ces échanges m'ont permis d'enrichir ma réflexion tout en ajoutant une dimension sociale à nos activités au laboratoire. Enfin, je remercie les membres du jury qui ont eu l'amabilité d'examiner ce mémoire : Professeur Gilles Pesant et Professeur Thomas Hurtut.

RÉSUMÉ

Le projet de recherche présenté dans ce mémoire vise à développer une nouvelle méthode de suivi d'objet sans modèle a priori à travers une séquence vidéo. À partir d'une position de la cible donnée dans la première trame, l'algorithme de suivi doit continuer à localiser cette cible dans chaque nouvelle trame. Cette fonction est requise dans de nombreuses applications : en vidéosurveillance pour déterminer des mouvements suspects, en robotique pour l'auto-localisation, ou encore en interface homme-machine comme la reconnaissance de gestes. Une large gamme de méthodes est proposée dans la littérature mais le suivi visuel d'objet reste un problème non-résolu. En effet, une grande variété de facteurs rend la tâche difficile, dont notamment la déformation de l'objet, le changement de ses couleurs dû aux variations d'illumination de son environnement, ou encore la présence d'éléments perturbateurs cachant l'objet.

Dans ce mémoire, nous proposons un algorithme de suivi qui représente la cible grâce à un modèle par parties. Cette représentation est adéquate pour gérer les déformations et les occultations. Si certaines parties de l'objet ne sont pas visibles ou non identifiables, d'autres permettront malgré tout de localiser la cible. Plus spécifiquement, nous définissons ces parties comme étant des superpixels modifiés par des points clés que nous appelons *SPiKeS (Superpixel-Keypoints Structure)*. Nous soutenons que ces deux types de caractéristiques sont complémentaires, le superpixel grâce à sa forme adaptée aux frontières des objets, le point clé grâce à sa discernabilité. En mettant leurs avantages à profit, nous obtenons un algorithme de suivi robuste à diverses situations complexes. L'expérimentation le prouve en montrant que notre méthode atteint des résultats compétitifs à de nombreuses méthodes de l'état de l'art. En outre, la supériorité de notre algorithme par rapport à des méthodes basées sur des superpixels ou des points clés uniquement, confirme l'intérêt de *SPiKeS*. Notons que ce dernier pourrait être exploité dans d'autres applications demandant une mise en correspondance efficace de régions d'intérêt.

ABSTRACT

This work introduces a model-free tracker robust to many challenging factors. A tracker is an algorithm dedicated to visual object tracking in a video. A model-free tracker has no information about the target except its position in the initial frame. The goal is to locate the target at each frame of the video. This task is required in a wide range of applications: in video surveillance for doubtful behaviors detection, in robotics for self-location, or in human-computer interaction such as gesture recognition. Numerous methods are proposed in the literature but visual object tracking is still an unsolved problem. Indeed, many challenging factors make the task difficult, including object deformations, illumination variations, rotations, viewpoint changes and occlusions.

In this thesis, we propose a tracker that represents the target with a part-based model. This kind of representation is well suited to handle deformations and occlusions. If some parts are hidden or unrecognizable, the others can still locate the target. More specifically, we define a part as a superpixel customized with keypoints, called a *SPiKeS (Superpixel-Keypoints Structure)*. We claim that these two features benefit from each other, the superpixel thanks to its boundary evidence, the keypoints thanks to their discriminativity. By leveraging a *SPiKeS*, we build a tracker that is capable of locating a target in many challenging situations. Experiments prove it by showing competitive performances of our tracker against the state-of-the-art. Above all, the superiority of our results compared to superpixels-only or keypoints-only trackers confirms the advantages of using *SPiKeS* inside a tracking procedure. Note that this new feature could be exploited in any other applications that would require an efficient image regions matching.

TABLE DES MATIÈRES

REMERCIEMENTS iii

RÉSUMÉ iv

ABSTRACT v

TABLE DES MATIÈRES vi

LISTE DES FIGURES viii

LISTE DES SIGLES ET ABRÉVIATIONS x

CHAPITRE 1 INTRODUCTION 1
1.1 Problématique 3
1.2 Objectif de recherche 7
1.3 Plan du mémoire 7

CHAPITRE 2 REVUE DE LA LITTÉRATURE 8
2.1 Modèle holistique 9
2.2 Filtre de corrélation 10
2.3 Modèle par parties 12
2.4 Suivi par détection 15
2.5 *Deep* tracker 16

CHAPITRE 3 DÉMARCHE DE L'ENSEMBLE DU TRAVAIL DE RECHERCHE . 20

CHAPITRE 4 ARTICLE 1 : SPIKES : SUPERPIXEL-KEYPOINTS STRUCTURE FOR ROBUST VISUAL TRACKING 27
4.1 Introduction 27
4.2 Background 30
4.3 Superpixel-Keypoints Structure 31
4.3.1 SPiKeS definition 31
4.3.2 SPiKeS comparison 31
4.4 The SPiKeS Tracker 33
4.4.1 Model 35
4.4.2 Matching 35

4.4.3 Location Estimation 36
4.4.4 Update 36
4.5 Experiments 38
4.5.1 Experimental setup 39
4.5.2 Evaluation 40
4.6 Conclusion 46

CHAPITRE 5 DISCUSSION GÉNÉRALE 50

CHAPITRE 6 CONCLUSION ET RECOMMANDATIONS 58

RÉFÉRENCES 59

LISTE DES FIGURES

Figure 1.1 Objectif du suivi : trouver à chaque trame t l'état S_t de l'objet localisé par S_0 dans la première trame I_0. 2
Figure 1.2 Différentes stratégies de recherche de la cible dans l'image. 4
Figure 1.3 Différents facteurs de difficulté pour le suivi d'objet. 6
Figure 2.1 Trois différentes représentations holistiques de la voiture encadrée en rouge. 11
Figure 2.2 Convolution entre une sous-image d'entrée et un filtre de corrélation ©2010 IEEE (Bolme et al., 2010). 11
Figure 2.3 Trois différents modèles par parties. 14
Figure 2.4 Différentes stratégies d'échantillonnage positif (vert) et négatif (rouge) ©2011 IEEE (Babenko et al., 2011). 17
Figure 2.5 Un classifieur est entraîné pour évaluer si un pixel est plutôt d'avant-plan ou d'arrière-plan ©2015 IEEE (Possegger et al., 2015). 17
Figure 2.6 Chaque couche localise la cible grâce à un filtre de corrélation. La position finale est déterminée par raffinement de chaque estimation ©2015 IEEE (Ma et al., 2015a). 18
Figure 3.1 Appariement des superpixels du modèle d'apparence (à gauche) et localisation par votes (flèches jaunes) dans la nouvelle trame. 20
Figure 3.2 Représentation simplifiée d'un modèle par superpixels. 21
Figure 3.3 Cas de déformation : fragments vs superpixels. 22
Figure 3.4 Cas d'occultation : les votes permettent aux superpixels de localiser le centre de l'objet contrairement à la maximisation du nombre de superpixels d'avant-plan dans le rectangle englobant. 23
Figure 3.5 Les superpixels appariés lors du suivi offrent une segmentation précise grâce à leur adhérence aux frontières de l'objet. 24
Figure 3.6 Complémentarité du superpixel et des points clés. 26
Figure 4.1 Decomposition of a frame into SPiKeS. Superpixels (black) structured by keypoints (red dot) linked by vectors (green). 29
Figure 4.2 SPiKeS representation. Keypoints are found inside or nearby the superpixel in a region of radius R around its center. Keypoints relative positions are given by vectors. 32

Figure 4.3 Tracking steps of our SPiKeS-based tracker. Keypoints detection (b) and superpixels segmentation (c) are processed on an input frame (a). Each superpixel forms a SPiKeS with its surrounding keypoints (d). Our SPiKeS model is matched with the new SPiKeS and the matching ones vote for the target's center (f) . The model is updated from the estimated bounding box (g) if no occlusion occurs. 34
Figure 4.4 A wrong vote of a background SPiKeS included in the model (cyan) will have a weak predictive factor ϕ. 39
Figure 4.5 A new keypoint (yellow, right) inside a matching superpixel (cyan) can be added to $\mathbf{K^f}$ because this superpixel belongs to the target, unlike the red keypoint. In the same way, a new superpixel (green, right) can be added to $\mathbf{S}^m$ because it includes a matching keypoint (white). . . . 40
Figure 4.6 Precision and Success plots for the one-pass evaluation (OPE) on OTTB. The number into brackets is the number of videos in the subset. . . . 42
Figure 4.7 Comparison of a superpixel-based tracker (DGT), a keypoint-based tracker (CMT) and ours (SPiKeS-T) on OTTB. 43
Figure 4.8 Influence of different superpixels-keypoints combinations on the overall performance on OTTB. 45
Figure 4.9 Qualitative results of top five trackers for sequences *bolt*, *woman*, *mountainBike*, *coke* from top to bottom. 45
Figure 5.1 Résultats de SPiKeS-T face au changement d'échelle. 48
Figure 5.2 Centre de la cible correctement localisé par quelques superpixels appariés (rouge) et différentes estimations de l'échelle. 49
Figure 5.3 Mise à l'échelle ajustée grâce à la classification de superpixels d'avant-plan. 49
Figure 5.4 La mise à l'échelle échoue due à une classification incorrecte. 50
Figure 5.5 Mise à l'échelle par paires d'appariement de points clés. 52
Figure 5.6 Séquence complexe *Ironman* où notre algorithme échoue à suivre sa cible. 53

LISTE DES SIGLES ET ABRÉVIATIONS

AS	Algorithme de Suivi
SPiKeS	Superpixel-Keypoints Structure
OR	Overlap Ratio
CLE	Center Location Error
CNN	Convolutional Neural Network
TPS	Trames Par Seconde
GPU	Graphics Processing Unit
CPU	Central Processing Unit

CHAPITRE 1 INTRODUCTION

Dans une vidéo, suivre une personne ou un objet consiste à déterminer sa position à chaque trame. Une application directe de ce concept permettrait à une caméra de suivre un sportif automatiquement. En vidéosurveillance, un système de suivi intelligent faciliterait l'analyse de déplacements d'une personne suspecte à travers de multiples caméras, allégeant le travail manuel d'un personnel de sécurité qui doit porter son attention sur plusieurs écrans simultanément. D'autre part, un algorithme de suivi est un outil essentiel qui s'intègre à de nombreuses applications. Citons notamment la reconnaissance de gestes qui nécessite tout d'abord de détecter l'instigateur du geste (un bras par exemple), puis de le suivre et analyser sa trajectoire pour au final déterminer l'action réalisée. On retrouve une procédure de suivi également dans les systèmes de réalité virtuelle. En effet, introduire un élément 3D dans une vidéo demande de connaitre la position de la caméra à chaque instant. Ce requis est calculé grâce à un suivi de points de repère présents dans la vidéo. Pour étendre son utilité jusqu'au domaine biomédical, lors d'une opération dans des zones difficiles d'accès à la vue du chirurgien, la localisation précise de l'outil permettra une meilleur visualisation de l'espace de travail et donc un meilleur contrôle.

Ces quelques exemples donnent un aperçu de la nécessité du suivi visuel d'objet, d'où l'activité de recherche importante dans ce domaine.

Ce mémoire s'intéresse plus particulièrement au problème du suivi visuel d'objet "unique" et "sans modèle a priori". "Unique", car contrairement au suivi "multiobjets" qui cherche à déterminer la trajectoire de plusieurs objets en même temps, une seule cible par séquence vidéo est considérée ici. "Sans modèle a priori", car la seule information disponible est un rectangle englobant la cible dans la première trame I_0 de la vidéo, comme illustré à la figure 1.1. L'objectif du suivi visuel est de trouver l'"état" S_t de cette cible dans les prochaines trames I_t. Le terme "état" réfère à la position de l'objet dans l'image ainsi qu'à la taille de celui-ci. Concrètement, cet état est modélisé par une boite/rectangle englobant dont le centre représente la position de l'objet et les dimensions du rectangle, sa taille.

Cette sorte de suivi où l'initialisation de la cible a lieu dans la première trame permet de suivre n'importe quel objet trouvé dans cette trame. Cette flexibilité est un avantage car cette catégorie d'algorithme ne sera pas spécialisée à suivre seulement un même type d'objet, comme des voitures, des animaux ou des humains. Cependant, puisqu'il n'y a aucune information a priori sur cette cible, les seuls indices que l'on peut en tirer sont dans la première trame à l'intérieur du rectangle englobant. Un être humain serait capable de reconnaitre la cible et

(a) Données (b) Inconnues

Figure 1.1 Objectif du suivi : trouver à chaque trame t l'état S_t de l'objet localisé par S_0 dans la première trame I_0.

savoir ce qu'il cherche car il est capable d'en inférer un modèle complet en trois dimensions, tandis qu'un ordinateur ne connaitra qu'une image de l'objet qu'il recherche. Imaginons que l'objet change de pose ou se déforme, son image sera différente de celle obtenue initialement. Comment savoir si c'est le même objet ou un objet différent ? Un autre scénario serait une cible cachée et un objet similaire apparaissant à un autre endroit de l'image. Un être humain est capable de savoir que l'objet est toujours caché tandis qu'un ordinateur voudra considérer cet objet comme celui à suivre puisqu'il est le même que dans la première trame.

Ces difficultés, parmi d'autres détaillées dans la section suivante, constituent la problématique de ce projet de recherche.

Le schéma global qu'adopte un algorithme de suivi visuel d'objet (AS) est le suivant :

1. **Initialisation**. Durant cette phase, l'algorithme de suivi crée un "modèle d'observation/apparence" de l'objet grâce à ce qu'il peut tirer d'information de son état connu dans la première trame. Son histogramme de couleurs est un exemple de modèle dit "holistique" car il représente l'objet dans son ensemble. D'autres modèles seront présentés dans le chapitre dédié à la revue de littérature.
2. **Suivi**. A chaque trame, l'AS cherche sa cible qu'il tentera de reconnaitre grâce à son modèle d'apparence. Des stratégies de recherche typiques sont les suivantes :
   - Recherche par "fenêtre glissante" : Toutes les positions de l'image sont testées, par conséquent aucune hypothèse sur le mouvement n'est considérée. Cette stratégie a l'avantage de détecter n'importe quelle amplitude de mouvement, à défaut d'être lente due à une recherche exhaustive. Une alternative est de considérer seulement

une zone de l'image autour de la position estimée dans la trame précédente, tel qu'illustré sur la figure 1.2a. Cependant, si la cible se trouve hors de cette zone, l'AS échouera.

- Recherche par "filtre à particules" : recherche statistique basée sur un "modèle de mouvement". Un nombre N d'états candidats aléatoires (particules) sont générés à partir de ce modèle de mouvement, ce qui limite alors la recherche à N zones différentes (voir figure 1.2b). Par exemple, un modèle de "marche aléatoire"

$$S_t = \mathcal{N}(S_{t-1}, \Sigma) \tag{1.1}$$

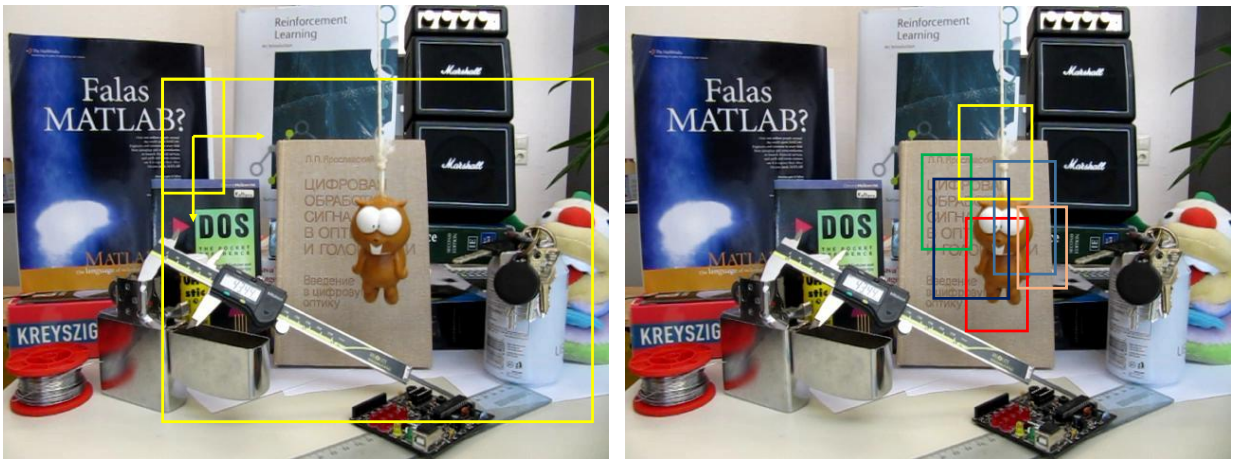
avec $\mathcal{N}(S_{t-1}, \Sigma)$ une distribution gaussienne, de moyenne égale à l'état de la cible dans la trame précédente et de matrice de covariance Σ. Intuitivement, une matrice de covariance de coefficients élevés autorisera un mouvement plus grand mais les particules seront plus espacées et le modèle de mouvement moins précis. D'un autre côté, augmenter le nombre de particules raffinerait la recherche à défaut d'en augmenter la complexité. Cette méthode est détaillée dans Isard and Blake (1998).

3. **Mise à jour du modèle d'apparence** : cette étape permet d'adapter et de compléter la représentation initiale. Imaginons que l'objet passe d'un endroit éclairé à une zone de plus en plus ombragée. Si le modèle se base sur les couleurs, il faudra les assombrir pour adapter le modèle à l'apparence qu'aura l'objet dans les trames futures. De même, une partie invisible de l'objet dans la première trame qui serait visible dans la trame courante est un nouvel élément essentiel à rajouter au modèle pour le compléter. Le seul moment pour obtenir ces nouveaux indices est pendant le suivi lui-même. Si celui-ci est incorrect, une mise à jour risque d'apporter de l'information erronée et donnera lieu à ce qu'on appelle la "dérive du modèle" : le modèle représente de moins en moins la cible initiale et l'AS commence à suivre autre chose.

On remarque que l'étape de suivi et de mise à jour sont fortement interdépendantes puisqu'une mauvaise mise à jour du modèle causera un suivi imprécis qui entraînera lui-même une mise à jour inadéquate.

1.1 Problématique

De nombreux facteurs rendent le suivi visuel d'objet une tâche ardue. Le critère de "robustesse" évalue la capacité d'un algorithme à suivre sa cible précisément dans n'importe quelle situation. Un AS peut être très précis dans certaines situations mais échouer dans une autre. Les difficultés sont autant liées au changement de forme de l'objet (déformation), au change-



(a) Fenêtre glissante dans une région restreinte (b) Filtre à particules

Figure 1.2 Différentes stratégies de recherche de la cible dans l'image.

ment de point de vue selon lequel il est observé, qu'à des modifications de son environnement (variation d'illumination, d'occultations) ou encore à des défis techniques propres à la séquence vidéo (mauvaise qualité). La figure 1.3 illustre certains des facteurs de difficulté qui sont présentés plus en détail dans la suite.

- **Déformation** : la déformation est un changement de géométrie. On en réfère également par le terme de "non-rigidité" de l'objet à suivre. Par exemple, une main est une cible qui se déforme selon qu'elle soit ouverte ou fermée, ce qui la rend difficile à modéliser à cause de ses apparences multiples et ce, même si le point de vue est identique. Un autre exemple est le tigre qui ouvre la bouche sur la figure 1.3a. Une mise à jour est cruciale pour pouvoir modéliser les différentes représentations et continuer à suivre l'objet.
- **Changement d'échelle** : mis à part la position du centre de la cible, un algorithme de suivi doit pouvoir déterminer sa taille, i.e. l'espace qu'elle occupe dans la vidéo. Un changement d'échelle a lieu lorsqu'elle s'éloigne ou se rapproche de la caméra (fig. 1.3b).
- **Rotation dans le plan** : l'apparence de la cible est identique mais elle a pivoté dans l'image (fig. 1.3c).
- **Rotation hors plan/changement de point de vue** : d'autres "facettes" de l'objet apparaissent. Sur la figure 1.3d, la canette de coca a pivoté, ce qui empêche de voir l'inscription qui était peut être un élément essentiel à la description du modèle. Ce problème est aussi considéré comme un changement d'apparence.
- **Basse résolution** : une vidéo en basse résolution sera moins détaillée qu'en haute

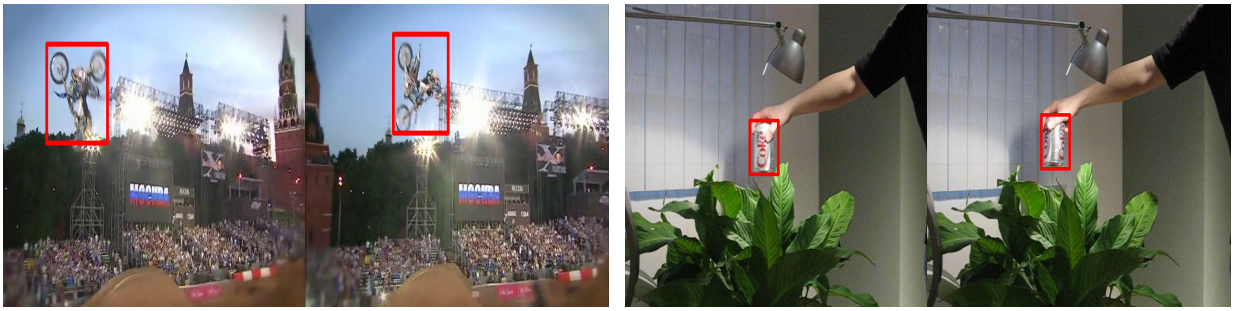
résolution. Par conséquent, représenter une cible à partir d'informations imprécises résultera en un modèle difficile à reconnaître dans la vidéo.

- **Flou de mouvement** : Ce phénomène change l'apparence de la cible, qu'elle soit rigide ou déformable. Il dépend du taux de trame, du mouvement de la caméra et de la vitesse de la cible. Le cerf de la figure 1.3e est difficilement discernable à cause du flou de mouvement.
- **Mouvement rapide** : Tout comme le flou de mouvement, ce problème provient d'un faible taux de trame ou d'un déplacement rapide de la caméra ou de la cible. Néanmoins, on ne parle pas ici de déformation mais au fait qu'une cible peut se trouver à des positions très éloignées entre deux trames consécutives. La majorité des algorithmes de suivi posent l'hypothèse de faible mouvement entre deux trames consécutives, ce qui permet notamment de limiter l'espace de recherche de la cible. Cependant, cette hypothèse n'est plus valide lorsqu'il y a un mouvement rapide.
- **Occultation** : la cible est cachée partiellement ou totalement par un élément de l'arrière-plan. Ce problème est un des plus complexes car l'AS doit localiser la cible à un endroit de l'image qui ne ressemble pas à la cible mais à l'objet occultant. Cependant, même si la localisation est correcte, il ne devrait pas modifier son modèle d'apparence au risque d'une dérive du modèle. Une solution serait de reconnaître les cas d'occultation. D'un autre côté, si la cible reste cachée pendant un très long moment et que la caméra se déplace, il faut pouvoir suivre l'objet occultant aussi puisque la cible se trouve en dessous.
- **Hors plan** : la cible n'est plus présente dans la vidéo. L'algorithme doit être capable de s'arrêter de la suivre et de la redétecter une fois qu'elle réapparait.
- **Arrière-plan bruité** : des éléments de l'arrière-plan sont ambigus par rapport à la cible, ils peuvent avoir la même texture ou des couleurs similaires. Certains éléments sont presque identiques à la cible comme sur la figure 1.3g où Usain Bolt (encadré) doit pouvoir être différencié des autres coureurs. On appelle aussi ces éléments perturbateurs des "distracteurs".
- **Variation d'illumination** : un objet passant d'une zone ombragée à une scène illuminée est un cas typique de changement d'illumination (fig.1.3h). L'apparence de l'objet varie, en particulier ses couleurs.

Outre sa capacité à être robuste, un algorithme de suivi doit aussi être performant, c'est à dire être assez rapide pour fonctionner en temps réel. En effet, si l'objectif du suivi est de repérer un déplacement suspect, il ne faudrait pas attendre une heure avant que la sécurité intervienne. Ce critère de vitesse n'est pas toujours évident à évaluer car il dépend de plusieurs composantes : la stratégie développée, le langage utilisé, l'ordinateur d'exécution,

(a) Déformation

(b) Changement d'échelle

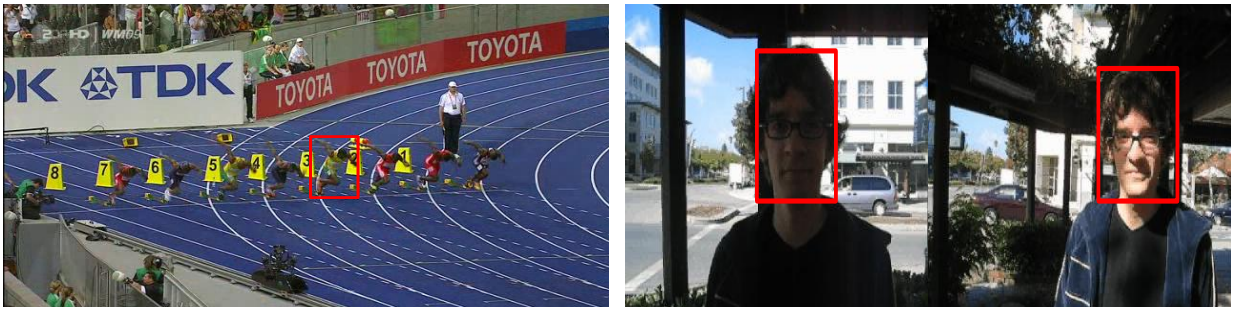


(c) Rotation dans le plan

(d) Rotation hors plan

(e) Flou de mouvement

(f) Occultation



(g) Distracteurs

(h) Variation d'illumination

Figure 1.3 Différents facteurs de difficulté pour le suivi d'objet.

l'optimisation du code, etc. En général, les méthodes plus précises seront plus lentes car elles auront tendance à développer des stratégies complexes afin de gérer un grand nombre de situations difficiles. Le compromis vitesse-précision sera donc souvent inévitable.

1.2 Objectif de recherche

Ce travail de recherche consiste à développer un nouvel algorithme de suivi, robuste aux différents facteurs cités plus haut, afin d'être capable d'évaluer précisément l'état d'un objet-cible à travers une vidéo. Plus spécifiquement, nous voulons :

- représenter la cible par un modèle reconnaissable dans des situations difficiles ;
- estimer précisément l'état de la cible dans chaque trame de la vidéo ;
- développer une stratégie de mise à jour compatible avec le modèle, capable de détecter les occultations pour éviter une dérive du modèle ;
- évaluer cette méthode sur un banc de vidéos-test reconnu par la littérature et la comparer à d'autres algorithmes de suivi de l'état de l'art.

1.3 Plan du mémoire

Le chapitre suivant présente une revue de littérature regroupant différents types d'algorithmes de suivi développés jusqu'à ce jour. La stratégie que nous avons décidé de suivre est expliquée dans le chapitre 3, introduisant ainsi notre travail de recherche présenté sous forme d'article de journal au chapitre 4. Enfin, la conclusion de ce mémoire présentera les limitations de notre méthode ainsi que les améliorations qui pourraient y être apportées.

CHAPITRE 2 REVUE DE LA LITTÉRATURE

Le but de ce chapitre est de fournir au lecteur un aperçu des différents types d'algorithmes de suivi que l'on peut trouver dans la littérature. Comme énoncé dans l'introduction, l'intérêt de ce domaine amène de nombreux chercheurs à développer de nouvelles méthodes toujours plus robustes et rapides. Bien que d'importants progrès ont déjà été réalisés, le suivi d'objet reste un problème complexe à résoudre étant donné la multitude de situations délicates mentionnées dans le chapitre 1. De manière générale, une méthode sera très précise dans certaines situations et moins dans d'autres. Certaines, essayant de pallier à toutes les difficultés, auront tendance à être trop complexes au dépens de la vitesse d'exécution. Ce compromis précision-vitesse est évidemment à prendre en compte en fonction de l'application visée, suivant que la contrainte de "temps réel" doive être respectée.

Il n'est pas aisé de catégoriser les algorithmes de suivi (AS), car ceux-ci peuvent employer différentes techniques qui les classifieraient autant dans une catégorie que dans une autre. Cependant, la littérature cite généralement deux grandes familles d'AS en fonction du modèle d'apparence/observation utilisé : les AS génératifs ou discriminatifs. Les AS à modèle génératif représentent la cible seulement à partir de l'information obtenue à l'intérieur du rectangle englobant qui la localise. Lors du suivi, ils recherchent dans chaque trame la région la plus similaire à ce modèle.

La deuxième famille d'AS, à modèle discriminatif, est apparue un peu plus tard avec le développement des techniques d'apprentissage machine. Le suivi est considéré comme un problème de classification de l'avant-plan (la cible) par rapport à l'arrière-plan. Dès lors, ce dernier est aussi mis à contribution dans l'apprentissage du modèle d'apparence.

L'une de ces familles est-elle meilleure que l'autre ? Encore faut-il pouvoir définir le terme "meilleur". Différents critères d'évaluation proposés dans la littérature sont utilisés par Kristan et al. (2014, 2015) pour classer différents AS. Selon les résultats de ces tests, il semblerait que les AS discriminatifs l'emportent. Cette victoire s'explique principalement par le fait que les modèles purement génératifs ne parviennent à se distinguer d'un arrière-plan trop complexe, amenant souvent à une dérive de leur modèle d'apparence. Par contre, les modèles discriminatifs ont besoin d'une grande quantité d'échantillons étiquetés afin d'obtenir un classifieur performant (Lasserre et al. (2006)). Dans le cas du suivi sans modèle à priori, obtenir ces échantillons est une tâche difficile car à l'initialisation, une seule trame est disponible. De plus, modifier un classifieur pour qu'il s'adapte aux changements d'apparence requiert encore plus d'échantillons. Ng and Jordan (2001) montrent au contraire que les modèles génératifs

n'ont besoin que de très peu d'échantillons pour obtenir une grande généralisation. Pour résumer, les modèles génératifs sont avantageux pour la facilité d'adaptation de leur modèle d'apparence tandis que les modèles discriminatifs permettront de mieux discerner la cible de l'arrière-plan. Profiter des avantages des deux modèles donne lieu à des méthodes "hybrides" (Zhong et al. (2012); Sui et al. (2015)), robustes dans de nombreuses situations, mais souvent très coûteuses en terme de calcul.

Afin d'enlever toute ambiguité, les différents AS présentés ci-dessous ne sont pas catégorisés explicitement dans une de ces deux familles, mais plutôt par type de représentation utilisée, celle-ci pouvant être autant générative que discriminative en fonction de l'utilisation des informations d'arrière-plan.

2.1 Modèle holistique

Un modèle holistique est une représentation de la cible dans son ensemble. Par exemple l'histogramme (de couleurs, de gradients, de contours, etc.) de la cible est une modélisation holistique (fig.2.1a). Un algorithme basé sur un histogramme de couleurs et combiné à une procédure "Mean-Shift" (Fukunaga (1990)) a été proposé par Bradski (1998). Le principe est de pondérer chaque pixel de la zone de recherche par sa probabilité obtenue dans l'histogramme. Le centre de la zone de recherche se déplace alors vers le centre de gravité de cette distribution et la procédure est réitérée avec le contenu de cette nouvelle zone jusqu'à convergence, définie comme un faible déplacement du centre entre deux itérations. En outre, la taille de la fenêtre de recherche est modifiée afin de s'adapter au changement d'échelle. Comaniciu et al. (2003) apportent une amélioration à cet algorithme en y ajoutant un noyau isotropique permettant de donner plus d'importance aux pixels proches du centre de la cible.

La modélisation par histogramme peut gérer certaines déformations si la distribution de couleurs de l'objet reste similaire. Par contre, en cas d'occultation, ce type de suivi ne pourra pas converger vers le centre de l'objet puisque l'histogramme d'un objet occultant n'est généralement pas identique à celui de la cible. De plus, la procédure de recherche "Mean-Shift" est limitée à la taille de la fenêtre de recherche, centrée à la position de l'objet dans la trame précédente. Par conséquent, il doit y avoir un recouvrement de l'objet entre deux trames consécutives, ce qui ne sera pas le cas si l'objet se déplace rapidement.

Dans l'article de Mei and Ling (2009), au lieu d'utiliser un histogramme, l'objet d'intérêt est représenté par une combinaison linéaire de patrons, aussi appelée représentation "éparse". Comme on peut le voir sur la figure 2.1b, ces patrons sont de simples images de la cible, obtenues pendant le suivi ou en décalant le rectangle englobant autour de la cible. Le suivi

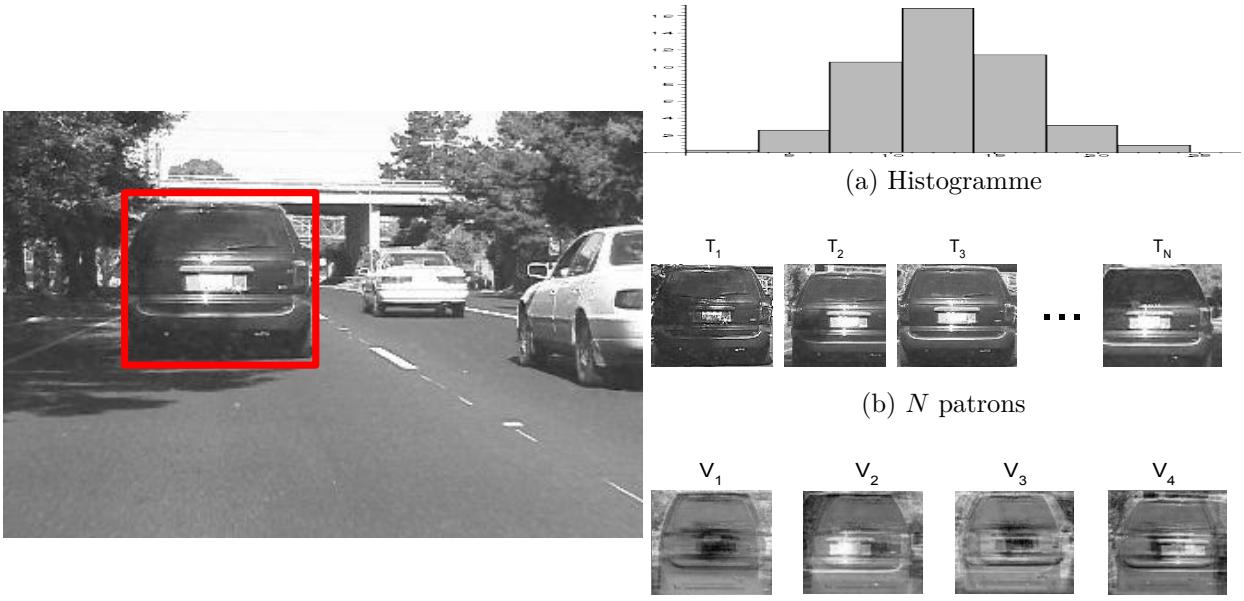
s'effectue par filtre à particules tel que décrit dans le chapitre précédent. Pour chaque sous-image déteminée par une particule, ses coefficients éparses sont calculés en minimisant un problème des moindres carrés régularisé l_1. La particule, d'erreur de représentation éparse la plus faible, est alors choisie comme étant le nouvel état de la cible. Les occultations sont prises en compte en introduisant un terme d'erreur dans la recherche des coefficients éparses et les changements d'illumination en ajoutant une contrainte de non-négativité sur ces coefficients.

Un autre modèle holistique possible est la représentation de la cible dans un "sous-espace". L'analyse en composantes principales (PCA) en est une méthode représentative que l'on retrouve notamment dans l'algorithme de suivi de Ross et al. (2008). La figure 2.1c montre les quatres premières composantes principales (appelées aussi axes principaux ou encore images "propres") obtenues après PCA. Contrairement à des patrons, ces composantes sont des images particulières rassemblant le plus de variance que la représentation de la cible pourrait avoir. L'erreur de reprojection d'une particule par ces axes principaux détermine l'état final si cette erreur est la plus faible. La motivation derrière cette représentation est de suivre la cible comme un "tout" plutôt que comme un ensemble de pixels indépendants. Plus récemment, Ma et al. (2015b) ont proposé une projection sur un sous espace non-linéaire afin de pouvoir gérer les déformations importantes de l'objet qu'un sous-espace linéaire ne pourrait pas représenter.

Enfin, une autre stratégie, proposée par Kwon and Lee (2010), combine plusieurs modèles holistiques de telle sorte que chaque modèle corresponde à une apparence spécifique de l'objet. En y ajoutant différents modèles de mouvement, cette approche est capable de prévoir autant un déplacement faible que rapide. Le tout forme une combinaison d'AS qui communiquent entre eux pour déterminer la localisation la plus précise de la cible.

2.2 Filtre de corrélation

Ce type de représentation est également holistique car un filtre de corrélation modélise l'apparence globale de la cible. Cependant, contrairement aux AS présentés plus haut qui étaient génératifs, un filtre de corrélation est appris à partir d'informations d'avant-plan et d'arrière-plan ce qui le range dans la famille des AS discriminatifs. En outre, leur principale caractéristique est leur rapidité d'exécution due à un traitement des données dans le domaine de Fourier. Le principe de fonctionnement est le suivant. La cible est localisée dans la nouvelle trame en convoluant une sous-image de celle-ci par un filtre. La réponse la plus élevée détermine la nouvelle position de la cible. La figure 2.2 illustre une convolution en deux dimensions. Calculer la convolution dans le domaine de Fourier limite la complexité à $\mathcal{O}(n \log n)$ à la place de $\mathcal{O}(n^2)$, avec n le nombre de pixels de la sous-image. Puisque le filtre constitue le modèle d'apparence de l'objet, il doit être appris à l'initialisation. L'algorithme MOSSE (Bolme et al.



(a) Histogramme

(b) N patrons

(c) 4 composantes principales

Figure 2.1 Trois différentes représentations holistiques de la voiture encadrée en rouge.

(2010)) apprend son filtre de réponse fréquentielle H en minimisant le problème des moindres carrés suivant :

$$\min_{H^*} \sum_{i}^{N} |F_i H^* - G_i|^2 \tag{2.1}$$

avec H^* le conjugué de H, F_i et G_i les transformées de Fourier des échantillons d'entrainement et de leur sortie respectivement. Dans cette méthode, les échantillons sont extraits dans les N premières trames.

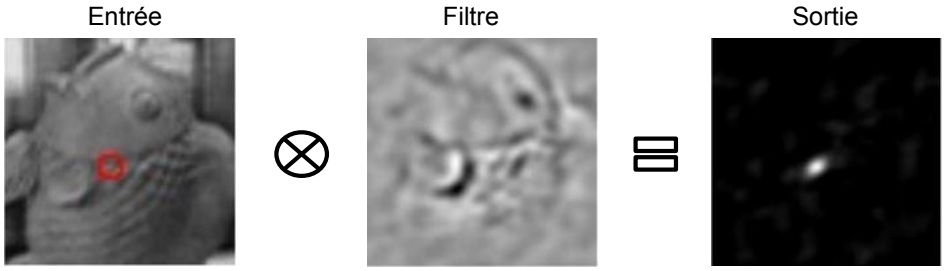


Figure 2.2 Convolution entre une sous-image d'entrée et un filtre de corrélation ©2010 IEEE (Bolme et al., 2010).

Contrairement à eux, Henriques et al. (2015) entrainent leur filtre seulement à partir de la première trame, en utilisant comme échantillons d'entrainement les décalages circulaires d'une sous-image centrée sur la cible. En effet, une transformée inverse d'un produit de deux

transformées de Fourier résulte en une convolution circulaire, cette dernière étant algébriquement caractérisée par une matrice circulaire. En formant cette matrice avec chaque décalage circulaire d'une sous-image centrée sur la cible, le filtre peut être appris rapidement dans le domaine de Fourier. En outre, au lieu de formuler le problème directement dans le domaine de Fourier, les coefficients du filtre sont déterminés par régression linéaire. En exploitant la structure circulaire des échantillons, ces coefficients sont calculés rapidement dans le domaine fréquentiel. De plus, en posant le problème comme une régression linéaire, ils tirent partie du "Kernel Trick" (Schölkopf and Smola (2002)) - astuce qui permet d'obtenir une régression non-linéaire en résolvant un problème linéaire - pour créer des filtres non-linéaires et modéliser plus fiablement leur cible. Enfin, contrairement à Bolme et al. (2010) qui n'utilisait que l'intensité des pixels, ils étendent le problème à plusieurs dimensions pour pouvoir utiliser des descripteurs plus robustes comme "HOG" (Felzenszwalb et al. (2010)), ou un autre espace de couleur tel qu'employé dans l'article de Danelljan et al. (2014a).

2.3 Modèle par parties

Les modèles holistiques, de par leur nature de représentation globale de l'objet, ne sont pas adaptés aux occultations et déformations. Certains d'entre eux sont capables d'être plus robustes, mais en proposant des stratégies souvent très complexes.

Les difficultés d'occultation et de déformation sont plus naturellement surmontées grâce à un "modèle par parties". Comme l'objet est décomposé en plusieurs parties, si certaines sont cachées, les autres peuvent toujours localiser la cible. Quant aux déformations, elles se traduiront par un déplacement des parties les unes par rapport aux autres, mais l'apparence de chacune d'elles restera la même. Dès lors, il devient plus simple de suivre chaque partie indépendamment et rassembler leurs positions pour localiser la cible entière.

Modèle par fragments

Un des premiers algorithmes de suivi dans cette catégorie est celui d'Adam et al. (2006). La cible est décomposée en fragments ("patchs" en anglais) rectangulaires répartis sur une grille uniforme (fig.2.3a). Chacun est représenté par son histogramme d'intensité. Dans une fenêtre de recherche centrée autour de la dernière position, chaque fragment du modèle votera pour une position dans cette fenêtre en fonction de sa similarité avec un fragment extrait à cette position. Tous les votes sont alors rassemblés pour déterminer la position finale de la cible. Pour parer un cas d'occultation, seulement les fragments les plus similaires ont le droit de voter. Cependant, cela assume un pourcentage de fragments toujours visible, ce qui ne sera

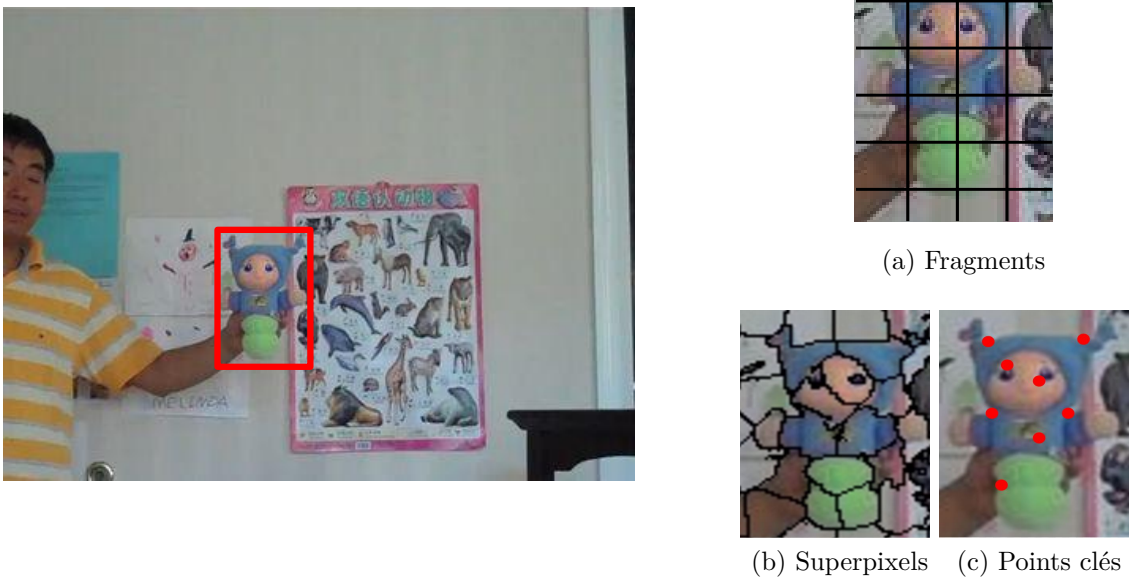
pas vrai en cas d'occultation totale. Ce modèle de fragments a été largement adopté par de nombreuses méthodes (Zhong et al. (2012); Jia et al. (2012); He et al. (2013)). Néanmoins, les objets de forme non-rectangulaire ne sont pas représentés adéquatement car des fragments rectangulaires incluront inévitablement des pixels d'arrière-plan, ce qui causera une dérive du modèle. Pour pallier à ce défaut, Li et al. (2015) assignent à chaque fragment un poids de telle sorte que ceux appartenant à l'arrière-plan n'affectent pas le suivi.

Modèle par superpixels

Pour mieux représenter un objet de forme non-rectangulaire, une solution serait d'adapter la forme des fragments en fonction de l'objet. Au lieu d'avoir des fragments rectangulaires, on aurait des fragments qui prennent la forme des contours de l'objet, comme sur la figure 2.3b. Dès lors, un fragment ne pourrait pas chevaucher deux objets différents, ce qui permettrait de ne pas avoir d'arrière-plan dans la représentation de l'objet. Ce type de fragment s'appellent aussi "superpixels", obtenu après "sur-segmentation" de l'image.

Un des instigateurs des AS basés sur les superpixels est Ren and Malik (2007). Après avoir sur-segmenté la trame en superpixels, chacun d'eux est étiqueté avant-plan ou arrière-plan. Les centres de gravité des superpixels d'avant-plan déterminent la position de la cible et la surface qu'ils occupent, la taille de la cible. Pour pouvoir étiqueter les superpixels dans la nouvelle trame, plusieurs composantes entrent en jeu. Tout d'abord, les superpixels sont mis en correspondance entre chaque trame en minimisant la "earth mover's distance" (similaire au problème de transport (Shi and Malik (2006)), basée sur la couleur moyenne et le déplacement du superpixel. Combiné à un modèle d'apparence (un histogramme de couleur de la cible et de l'arrière-plan) au sein d'un champ aléatoire conditionnel, les superpixels de la nouvelle trame peuvent être alors classifiés. Malgré sa capacité à gérer les déformations, cette méthode ne propose pas de stratégie contre l'occultation ni pour les variations d'illumination. En outre, elle souffre d'une complexité de calcul importante due à la génération des superpixels par triangulation de Delaunay.

L'algorithme de suivi "Superpixel Tracker" de Wang et al. (2014) emploie plutôt une segmentation de complexité linéaire SLIC (Achanta et al. (2012)). Catégorisé dans la famille des AS discriminatifs, celui-ci attribue à chaque superpixel une probabilité de faire partie de l'objet. Cette probabilité est calculée en fonction de sa distance par rapport aux "agrégats" ("clusters" en anglais) constituant le modèle discriminatif. Ce modèle est appris durant les premières trames de la vidéo, où des superpixels d'avant-plan et d'arrière-plan sont extraits pour apprendre les clusters. Par conséquent, un autre algorithme de suivi ou une assignation manuelle est requise, ce qui est un des défauts de cette méthode. De plus, puisque la recherche



(a) Fragments

(b) Superpixels (c) Points clés

Figure 2.3 Trois différents modèles par parties.

du nouvel état est limitée à une fenêtre où sont extraits les états candidats, cet AS ne peut suivre des déplacements importants.

Au lieu de classifier les superpixels, Wang and Yagi (2014) préfèrent une approche générative. Chaque superpixel composant leur modèle est mis en correspondance avec ceux de la nouvelle trame. En décrivant le superpixel par sa couleur moyenne, qui n'est pas un descripteur très discriminatif, ils autorisent plusieurs correspondances pour un même superpixel et calculent le déplacement par rapport à chaque correspondance. En rassemblant les déplacements de chaque superpixel, ils obtiennent une carte de déplacement où le déplacement commun à tous les superpixels aura plus de poids dans cette carte, fournissant alors la position finale de la cible. Par contre, aucune méthode de mise à l'échelle n'est proposée.

A la place d'autoriser plusieurs correspondances, Cai et al. (2014) tirent parti de la position des superpixels les uns par rapport aux autres, en créant un modèle génératif représenté par un graphe de superpixels. Dès lors, en mettant en correspondance ce graphe plutôt que les superpixels individuellement, ils obtiennent une correspondance moins ambigue pour chacun des superpixels, même si ceux-ci sont seulement décrits par leur couleur moyenne. La position finale est déterminée par le vote de chacun de ces superpixels pour le centre de la cible. Néanmoins, la technique de "correspondance spectrale" requiert des hypothèses contraignantes pour pouvoir être appliquée en un temps raisonnable. De plus, une autre

composante de cet alogrithme est une classification SVM (V. Vapnik (1995)) sommaire afin de limiter les superpixels à mettre en correspondance. Cependant, si la classification échoue, des superpixels d'avant-plan seront inévitablement omis.

Modèle par points clés

Parmi les modèles par parties, on trouve aussi les modèles reposant sur des "points clés" tels que SIFT (Lowe (2004)), SURF (Bay et al. (2006)) ou encore BRISK (Leutenegger et al. (2011)). Ces descripteurs de bas niveau, largement utilisés tant en classification qu'en reconnaissance et détection d'objet, ont l'avantage d'être plus discriminatifs que les superpixels et invariants à certaines transformations (Mikolajczyk and Schmid (2005), Tuytelaars and Mikolajczyk (2008)). Dans le domaine du suivi visuel, Nebehay and Pflugfelder (2014) utilisent un modèle génératif de points clés (fig. 2.3c). A chaque nouvelle trame, des points clés sont extraits et mis en correspondance avec ceux du modèle. Ces correspondances permettent alors de voter pour la position du centre de l'objet. Bouachir and Bilodeau (2015) reprennent cette idée en pondérant les points clés du modèle en fonction de leur confiance, i.e. s'ils votent correctement et fréquemment pour le centre de l'objet, permettant ainsi d'éviter que des points clés d'arrière-plan n'affectent le suivi. De par sa représentation locale, ce type de descripteur est naturellement robuste aux déformations et occultations. De plus, certains types de points clés sont également invariants aux rotations et changements d'échelle. Par contre, si peu ou aucun point clé n'est détecté, par exemple lorsque l'objet est peu texturé, la représentation du modèle fait défaut et le suivi sera imprécis.

2.4 Suivi par détection

Les algorithmes de suivi de cette catégorie sont typiquement de la famille discriminative. L'idée sous-jacente du suivi par détection est l'utilisation d'un classifieur qui permet d'évaluer si une région de l'image correspond à l'objet à repérer. Cette classification peut-être binaire ou continue, montrant la probabilité qu'un pixel appartienne à la cible comme sur la carte de confiance de la figure 2.5b. Un des premiers AS à combiner suivi et détection est SVT (Avidan (2004)). Celui-ci pré-entraine un classifieur SVM spécialisé dans la détection de véhicule. Ensuite, à partir d'un état candidat initialisé à la position estimée de la trame précédente, il cherche la translation qui maximise le pointage donné par le classifieur SVM. Mis à part le fait que cet AS ne puisse gérer les cas d'occultation, la limitation la plus importante est celle liée au suivi d'un seul type d'objet. Si l'on veut avoir un classifieur propre à un objet quelconque déterminé dans la première trame de la vidéo, il faudrait entrainer ce classifieur "en ligne", i.e. pendant le suivi. Cela requiert de sélectionner des échantillons positifs et

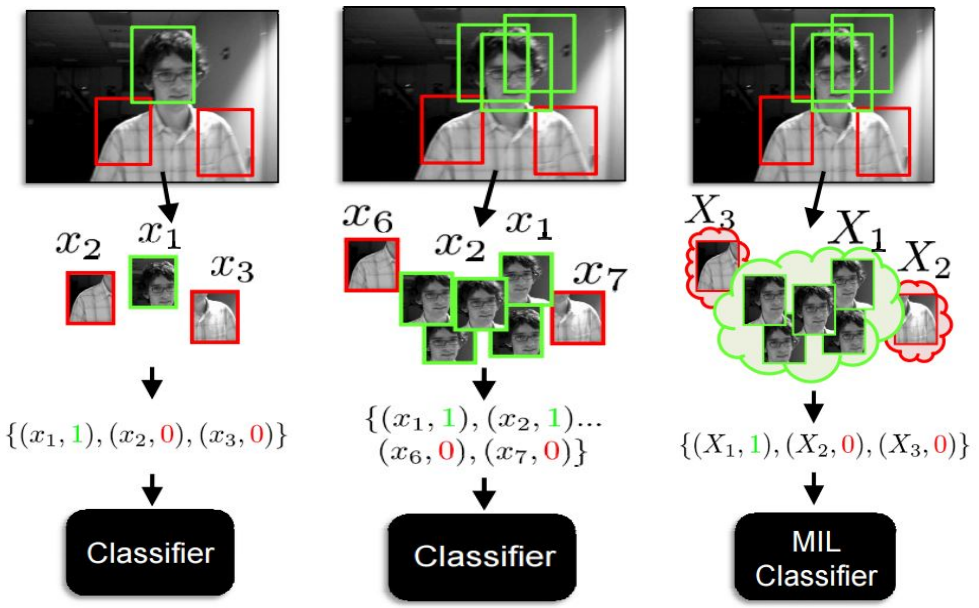
négatifs au fil de la vidéo mais lesquels choisir ? Et combien ? La figure 2.4 illustre différents échantillonnages possibles. L'image 2.4a sélectionne un échantillon positif (vert) et plusieurs échantillons négatifs (rouge). Ne prendre qu'un seul échantillon positif pour entrainer le classifieur donnera une classification trop discriminative, surtout si cet échantillon positif ne représente pas correctement la cible (e.g. seulement une partie de la cible est comprise dans cet échantillon). Dans le cas suivant (fig. 2.4b) plusieurs échantillons positifs sont sélectionnés autour de la cible, mais tous ne représentent pas adéquatement la cible (e.g. de l'arrière-plan inclus). Par conséquent, le classifieur ne parviendra pas à différencier précisément l'avant-plan de l'arrière-plan. On note ici la difficulté de devoir choisir des échantillons pour entrainer au mieux un classifieur.

L'AS MIL (Babenko et al. (2011)) propose une solution utilisant un "apprentissage par multiples instances". Le classifieur est entrainé non pas avec les échantillons eux-mêmes comme sur la figure 2.4b mais plutôt avec des "sacs d'instances/échantillons" (fig. 2.4c). Puisque dans le sac d'instances positif, au moins un échantillon représentera correctement la cible, celui-ci aura plus de poids dans l'apprentissage du classifieur pour le rendre plus adéquat à la cible, contrairement au cas 2.4b où chacun était traité comme échantillon positif indépendant, même si certains correpondaient moins à la cible que d'autres.

L'AS de Hare et al. (2011) pallie le problème du choix des échantillons en combinant la classification et le suivi dans une même formulation. Un état candidat est classifié non pas seulement en fonction de son apparence mais aussi par rapport à la position qu'il occupe dans la nouvelle trame. Par conséquent, la translation et l'apparence d'un candidat constitue un échantillon qui sera évalué par le classifieur. En formulant le problème de cette manière, l'échantillon de pointage maximum sera pris comme nouvel échantillon positif et celui de pointage minimum comme échantillon négatif. Contrairement à un filtre à particules, la zone de recherche est un disque centré à la position précédente.

2.5 *Deep* tracker

La technique d' "apprentissage profond" ("deep learning" en anglais) utilisant des "convolutional neural networks"(CNN) s'est révélée une technique d'apprentissage machine très efficace dans de nombreux domaines, notamment en segmentation (Long et al. (2015)) et classification d'image (Krizhevsky et al. (2012)), ou encore pour la détection d'objet (Girshick et al. (2012)). Dès lors, de très récents algorithmes de suivi l'ont adoptée pour créer un modèle d'apparence robuste. L'attrait principal est de remplacer les descripteurs typiques (HOG, points clés, arêtes,etc.) par des descripteurs adaptés à la tâche requise. Par exemple, MDNet (Nam and Han (2015)) pré-entraine son réseau de neurones sur des séquences vidéo

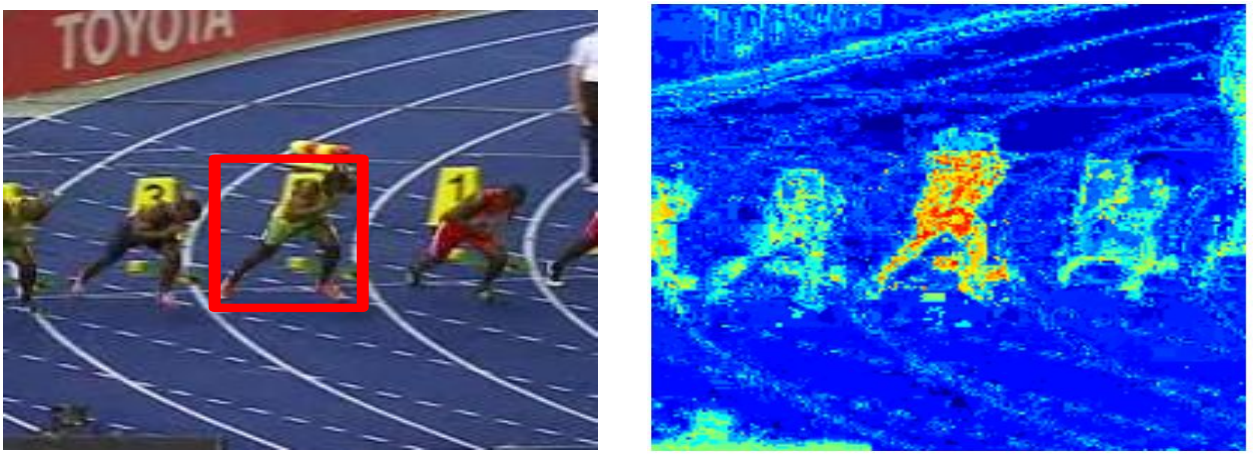


(a) Unique échantillon positif pris comme étant l'état estimé.

(b) Plusieurs échantillons positifs pris autour de l'état estimé.

(c) Unique échantillon positif considéré comme un "sac d'instances".

Figure 2.4 Différentes stratégies d'échantillonnage positif (vert) et négatif (rouge) ©2011 IEEE (Babenko et al., 2011).



(a) Avant-plan à l'intérieur du rectangle rouge.

(b) Carte de probabilité des pixels d'avant-plan (vers le rouge) et d'arrière-plan (vers le bleu).

Figure 2.5 Un classifieur est entraîné pour évaluer si un pixel est plutôt d'avant-plan ou d'arrière-plan ©2015 IEEE (Possegger et al., 2015).

spécifiques au suivi visuel d'objet afin d'obtenir une représentation générique holistique d'une cible quelconque. Un deuxième entrainement est réalisé pendant le suivi pour adapter le réseau à la cible choisie dans la première trame. Après avoir généré des états candidats dans la nouvelle trame par filtre à particules, l'état choisi sera celui qui aura obtenu le plus grand pointage après être passé dans le CNN. Une autre méthode pour localiser un objet recourant à un CNN est proposée par Ma et al. (2015a). Lorsqu'on avance dans le réseau, les couches donnent une représentation de plus en plus invariante mais, de par leur faible résolution, ne permettent pas de localiser précisément la cible. Dès lors, un filtre de convolution, comme expliqué plus haut, est associé à chaque couche. Cela permet d'obtenir une localisation de la cible qui est raffinée de plus en plus lorsque l'on remonte dans les couches comme illustré sur la figure 2.6.

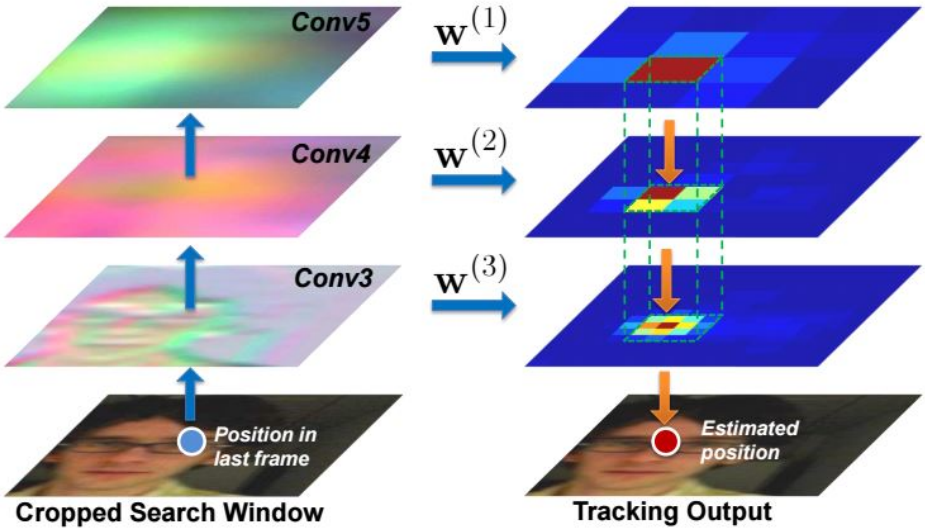


Figure 2.6 Chaque couche localise la cible grâce à un filtre de corrélation. La position finale est déterminée par raffinement de chaque estimation ©2015 IEEE (Ma et al., 2015a).

Selon le concours VOT2015 (Kristan et al. (2015)), les deux algorithmes de suivi les plus précis, mais aussi les moins performant en terme de vitesse, reposent sur des CNN. L'avantage principal réside dans les caractéristiques extraites pour représenter la cible. Grâce à l'entrainement d'un CNN sur une grande base de donnée, une cible pourra être représentée par des caractéristiques adaptées à celle-ci, i.e. elles permettront de reconnaitre la cible dans diverses situations (déformation, occultation, etc.). D'ailleurs, l'étude réalisée par Wang et al. (2015) démontre que le choix des caractéristiques pour représenter le modèle d'apparence est le facteur le plus déterminant quant à la précision d'un algorithme de suivi. Par contre, le nombre élevé de convolution augmentant avec le nombre de couches du réseau rend la mé-

thode coûteuse en terme de calculs. Une implémentation sur GPU est souvent requise pour exécuter ce type de méthode en un temps raisonnable.

Durant ce chapitre, différentes méthodes de suivi ont été présentées, chacune d'elles ayant des points forts et des points faibles. Dans ce travail, la direction que nous avons choisi de suivre est un modèle par parties basé sur des superpixels qui seront dédiés à voter pour la position de l'objet dans la nouvelle trame. Le chapitre suivant explique la motivation de cette stratégie.

CHAPITRE 3 DÉMARCHE DE L'ENSEMBLE DU TRAVAIL DE RECHERCHE

Parmi les types d'algorithmes de suivi présentés dans le chapitre précédent, la stratégie que nous avons décidé de suivre dans ce travail de recherche est la suivante. Le modèle d'apparence sera un modèle par parties, en l'occurence, des superpixels. Chacun d'eux connait sa position par rapport au centre de l'objet. Dès lors, lorsqu'un superpixel du modèle est mis en correspondance dans une nouvelle trame, il peut "voter" pour le centre de l'objet. Cette idée est illustrée à la figure 3.1.

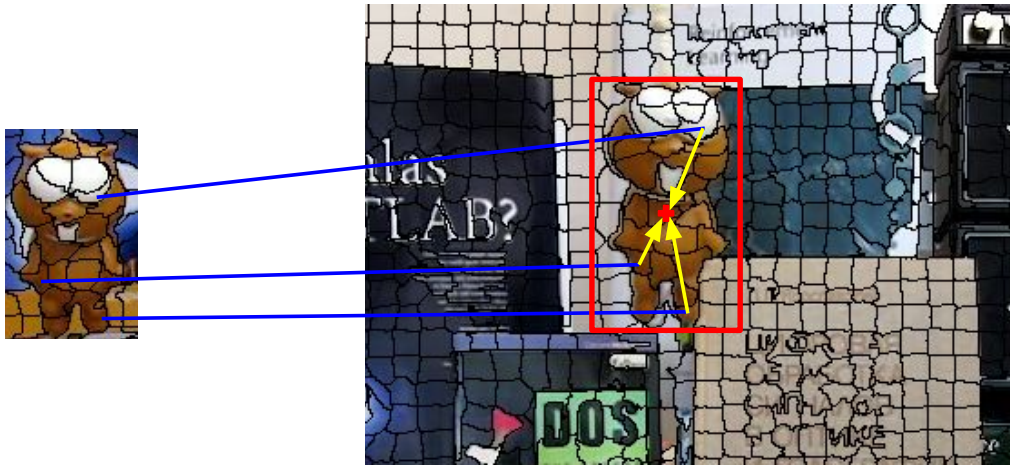
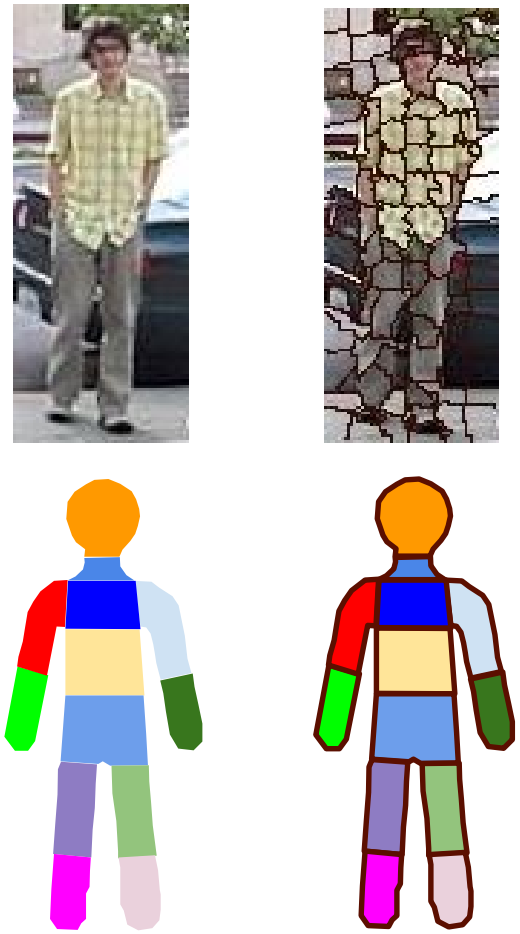


Figure 3.1 Appariement des superpixels du modèle d'apparence (à gauche) et localisation par votes (flèches jaunes) dans la nouvelle trame.

Pour comprendre la motivation du choix d'un modèle par superpixels, illustrons leurs avantages par des exemples. La figure 3.2 montre une segmentation d'un humain en superpixels. On parle aussi de sur-segmentation car en général, plusieurs superpixels segmentent un même objet. Afin de faciliter la compréhension des explications qui vont suivre, utilisons plutôt le bonhomme multicolore de la figure 3.2a segmenté en superpixels (fig. 3.2b). Notons que cet exemple non-réaliste, dont la segmentation en superpixels est idéale, n'est présenté que pour appuyer l'idée d'utiliser avantageusement les superpixels. Nous verrons plus loin ce qu'il en est pour des cas réels.

Tout d'abord, contrairement à un modèle holistique, le fait d'être un modèle par parties le rend plus apte à être reconnaissable lors de déformations géométriques locales. Si l'on observe l'exemple de la figure 3.3b, les représentations globales sont fort différentes (jambes



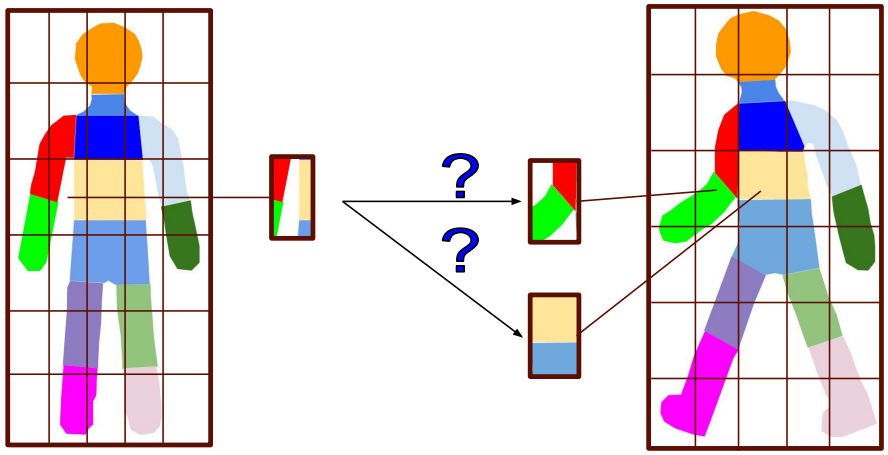
(a) Cible. (b) Cible segmentée en superpixels.

Figure 3.2 Représentation simplifiée d'un modèle par superpixels.

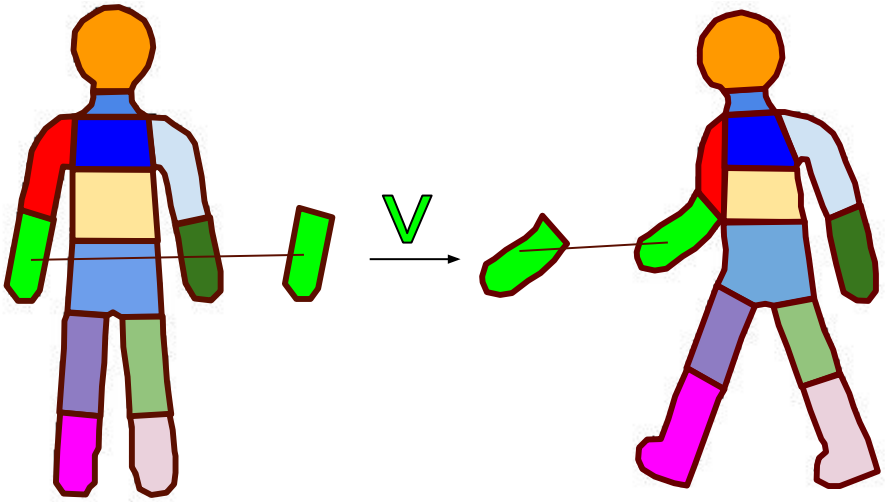
plus espacées, bras plié), alors que chaque partie (représentée par un superpixel) est facilement reconnaissable d'une représentation à l'autre, autant par sa couleur que par sa forme.

Deuxièmement, la forme des superpixels s'adapte à l'objet. Leur contenu sémantique est plus cohérent qu'un simple fragment rectangulaire. En effet, contrairement à ces derniers, un superpixel idéal, par définition, ne chevauchera pas deux objets différents, alors qu'un fragment pourrait contenir de l'information de différentes parties. Cet inconvénient, illustré sur la figure 3.3a, rend l'appariement de fragments ambigu alors qu'il l'est moins pour les superpixels (fig. 3.3b).

Ensuite, il a été énoncé dans la revue de littérature que les modèles par parties étaient robustes dans des situations d'occultation. Effectivement, puisqu'une partie de la cible est cachée, son apparence globale est fortement modifiée. Par contre, l'apparence des superpixels visibles



(a) Ambiguité d'appariement d'un fragment rectangulaire.

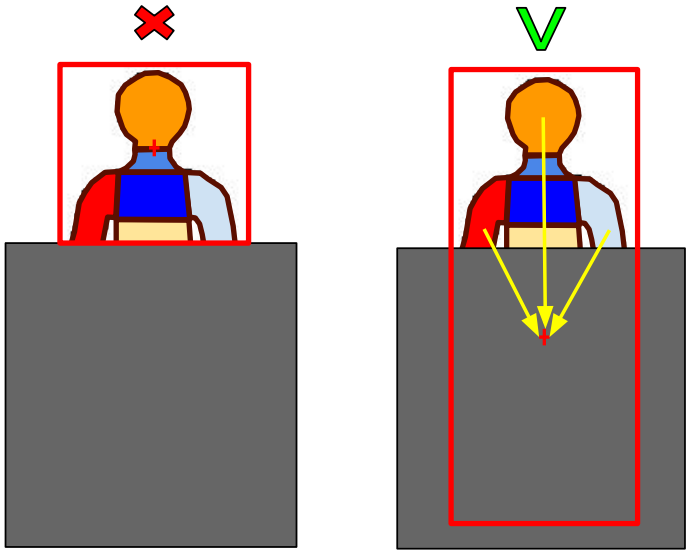


(b) Appariement évident d'un superpixel.

Figure 3.3 Cas de déformation : fragments vs superpixels.

reste assez similaire, ce qui permet de les identifier plus aisément. La question maintenant est de savoir comment localiser la cible à partir de ces superpixels appariés au modèle. Un choix possible serait de prendre le rectangle englobant le nombre maximum de superpixels mis en correspondance, comme illustré sur la figure 3.4a. Dans cette situation, le centre de la cible est erroné car il est positionné près du visage alors qu'il correspond normalement au centre du corps. La situation de la figure 3.4b est donc plus correcte : le centre est localisé par des flèches jaunes qui correspondent à ce qu'on appelle les "votes" des superpixels.

En possédant chacun un vote, les superpixels du modèle d'apparence encodent la structure de l'objet car chacun est positionné par rapport au centre de l'objet grâce à ce vote. Dès lors, en transférant ce vote à leurs correspondances trouvées dans une nouvelle trame, celles-ci



(a) Localisation de l'objet par maximisation du nombre des superpixels d'avant-plan.

(b) Localisation de l'objet par vote.

Figure 3.4 Cas d'occultation : les votes permettent aux superpixels de localiser le centre de l'objet contrairement à la maximisation du nombre de superpixels d'avant-plan dans le rectangle englobant.

sont capables de localiser le centre de l'objet, autrement dit, sa position globale.

Enfin, bien que la segmentation vidéo ne soit pas le sujet de ce travail, notons l'avantage que peut apporter un suivi par superpixel dans ce genre d'application. Après un suivi, les superpixels détectés comme avant-plan (i.e. les appariements) donnent une segmentation précise de la cible (fig. 3.5), notamment grâce à leur adhérence à la forme de l'objet. Une idée similaire a d'ailleurs été exploitée par Wen et al. (2015).

En réalité, les objets à suivre ne sont pas aussi simples que le bonhomme multicolore de la figure 3.2, dont chaque superpixel est de couleur différente. Au contraire, plusieurs superpixels auront souvent la même couleur et une forme similaire. Ces deux types de caractéristiques ne sont alors pas assez discriminantes pour assurer un correct appariement. La figure 3.6a montre une mise en correspondance erronée due au manque de discriminativité des couleurs. Dès lors, rendre ces superpixels plus discernables les uns des autres est nécessaire pour pouvoir mettre au point la stratégie de suivi planifiée, i.e. la localisation par votes après mise en correspondance.

L'idée que nous proposons dans l'article du chapitre suivant est un superpixel amélioré par des points clés. Comme discuté dans la revue de littérature, les points clés sont des points

Figure 3.5 Les superpixels appariés lors du suivi offrent une segmentation précise grâce à leur adhérence aux frontières de l'objet.

facilement discernables et invariants à certaines transformations affines. Ils pourraient ainsi enlever l'ambiguité entre deux superpixels candidats de couleur similaire, telle qu'illustrée sur la figure 3.6b. Notons que les points clés ont été représentés par des disques colorés seulement pour les discerner plus facilement dans ces exemples. En réalité, leur description est autre qu'une unique couleur et dépend de la méthode utilisée. Par exemple, les points SIFT de Lowe (2004) sont robustes aux changements d'illumination, invariants aux rotations et reconnaissables à différentes échelles. Pour les extraire d'une image, il faut convoluer celle-ci avec des noyaux gaussiens de différents écart-types. Ensuite, ces convolutions sont soustraites les unes aux autres, ce qui donne lieu à des images de "différences de gaussiennes". Enfin, les optima locaux trouvés sur ces dernières sont les points clés. Afin de pouvoir les comparer, chacun d'eux requiert un descripteur. Pour les points SIFT, le descripteur est un histogramme d'orientation des gradients situés aux alentours du point clé. La détection et la description de caractéristiques, telles que le sont les points clés, est un sujet de recherche toujours très actif. Nous dirigeons le lecteur vers ces articles (Lowe, 2004; Mikolajczyk and Schmid, 2005; Bay et al., 2006; Tuytelaars and Mikolajczyk, 2008; Leutenegger et al., 2011) pour plus de détails sur les différentes méthodes existantes.

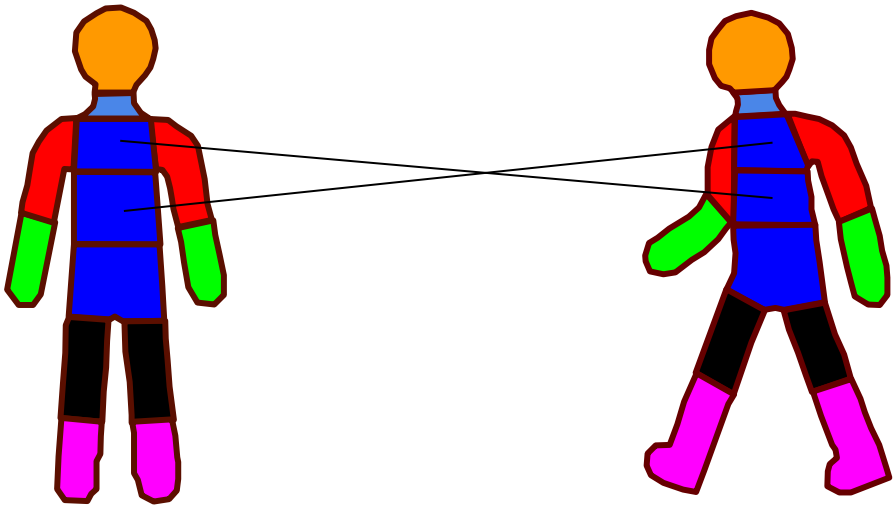
Le lecteur pourrait se demander : pourquoi ne pas utiliser que des points clés ? Le nombre de points clés détectés varie d'une image à l'autre. Il est possible d'en avoir beaucoup ou au contraire, aucun. Un objet peu texturé ou de couleur uniforme n'aura en général que très peu de points clés. Par conséquent, une localisation basée sur seulement quelques points clés aura plus de chance de faillir si certains d'entre eux sont mal appariés. Par contre, une image peut toujours être décomposée en un même nombre fixe de superpixels, assurant ainsi d'en trouver dans la nouvelle trame. En outre, quelques points clés seulement peuvent rendre discriminants plusieurs superpixels.

Enfin, le chapitre précédent introduisait une étape de mise à jour du modèle afin de le

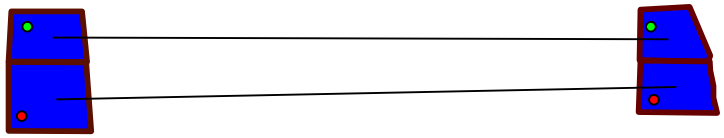
compléter au fur et à mesure du suivi. Si l'on veut rajouter un nouveau point clé au modèle, il faut qu'il fasse partie de l'objet et non pas de l'arrière-plan. Grâce à la détection des superpixels d'avant-plan, si un point-clé se trouve à l'intérieur d'un de ceux-ci, alors ce point clé est sûr de faire partie de l'objet et peut donc être rajouté au modèle (fig. 3.6c).

Dès lors, nous revendiquons que les superpixels et les points clés sont deux caractéristiques complémentaires. Leur fusion peut être mise à profit au sein d'une procédure de suivi qui serait alors robuste à diverses situations difficiles.

L'article qui suit détaille la manière dont nous avons assemblé les superpixels et les points clés pour créer un nouveau descripteur, ainsi que leur mise en correspondance et leur intégration au sein d'une procédure de suivi. Le résultat final est un nouvel algorithme de suivi dont la précision est compétitive aux méthodes de l'état de l'art.



(a) Une similarité trop grande entre deux superpixels cause un mauvais appariement.



(b) Des superpixels similaires sont discernables grâce aux points clés (points colorés).

(c) Quel point clé entre le jaune et le rose peut-on rajouter au modèle ? Le point clé jaune car il est à l'intérieur du superpixel apparié, donc de la cible.

Figure 3.6 Complémentarité du superpixel et des points clés.

CHAPITRE 4 ARTICLE 1 : SPIKES : SUPERPIXEL-KEYPOINTS STRUCTURE FOR ROBUST VISUAL TRACKING [1]

Abstract

In visual tracking, part-based trackers are attractive since they are robust against occlusion and deformation. In contrast, a part represented by a rectangular patch does not account for the shape of the target, while a superpixel does thanks to its boundary evidence. Nevertheless, tracking superpixels is difficult due to their lack of discriminative power. Therefore, to enable superpixels to be tracked discriminatively as object parts, we propose to enhance them with keypoints. By combining properties of these two features, we build a novel element designated as a Superpixel-Keypoints structure (SPiKeS). Being discriminative, these new object parts can be located efficiently by a simple nearest neighbor matching process. Then, in a tracking process, each match votes for the target's center to give its location. In addition, the interesting properties of our new feature allows the development of an efficient model update for more robust tracking. According to experimental results, our SPiKeS-based tracker proves to be robust in many challenging scenarios by performing favorably against the state-of-the-art.

4.1 Introduction

A robot needs to track its target to interact with it. Doubtful behaviors can be detected thanks to tracking in visual surveillance. Hands are tracked for gesture recognition. Those examples show only a small part of a wide range of tracking applications, thus encouraging many research efforts to focus on this topic. When no prior information about the object to track is available, tracking is referred to as model-free tracking. In a video, the goal is to locate a particular target given its location only in the first frame. This task is challenging because of numerous factors such as illumination variation modifying object color, occlusion hiding some parts, or new parts appearing if the viewpoint changes. While some of these issues are handled efficiently by different techniques, it is challenging for a single tracker to handle them all.

Trackers are generally split into two categories : discriminative and generative. Discriminative trackers consider tracking as a binary classification problem. Samples from foreground and background are selected to train a classifier that is able to separate the target from the rest

1. F.-X. Derue, G.-A. Bilodeau et R. Bergevin, "SPiKeS : Superpixel-Keypoints Structure for Robust Visual Tracking", article soumis à la revue IEEE Transactions on Image Processing , mars 2016.

of scene. Afterwards, this target detection yields a location estimation. This is the typical "tracking-by-detectio" framework followed by many discriminative trackers (Avidan, 2004; Bai et al., 2013; Babenko et al., 2011; Zhou et al., 2011), although Struck (Hare et al., 2011) achieves the classification and location in one step. In most approaches, samples are selected randomly, limiting their number for computational efficiency. Instead of random samples, Henriques et al. (Henriques et al., 2012) proposed to select all the samples and exploited their redundancy to build a kernel classifier which tracks very quickly in the Fourier domain. Due to its simplicity and rapidity, many recent trackers build upon it (Danelljan et al., 2014a; Henriques et al., 2015; Li et al., 2015).

In generative trackers, only foreground information models the target appearance and the tracking task aims to find the most similar image region to this model. In (Mei and Ling, 2009), the target is represented with a sparse model by using templates. The tracking location is the patch whose projection error into the template space is minimum. To account for appearance change and different kinds of motion, Kwon et al. (Kwon and Lee, 2010) build different observation and motion models, so that each pair can be used within a basic tracker. These multiple basic trackers are then integrated into a main tracker, which is more robust thanks to the interaction between its components. Although those methods can handle some appearance alterations, they are not robust against deformation and occlusion due to their holistic representation. These issues are usually handled by the family of part-based trackers. As the model is decomposed into several parts, an occlusion only affects some of them, without preventing the other ones to track the target. Typically, usual approaches consider the parts as rectangular patches structured in a grid (Adam et al., 2006; Zhong et al., 2012; Jia et al., 2012; He et al., 2013). However, non-rectangular targets are not well represented because background patches inside the bounding box inevitably affect the model and make it drift. To this end, Li et al. (Li et al., 2015) assigns reliability to patches so that noisy background patches do not affect the tracking.

Another part-based approach consists of oversegmenting the target into superpixels. Thanks to their boundary evidence, they take better the shape into account. In (Wang et al., 2014), a map is built showing the probability of a superpixel to belong to the target and the target location is the area with maximum likelihood. This tracker shows good performance but it needs a model which has to be learned in the first frames. Therefore, it needs manual annotation or another tracker for the initialization step. Recent approaches such as (Cai et al., 2014) and (Wang and Yagi, 2014) propose to integrate superpixels in a matching-tracking framework. An appearance model is built with superpixels and each of them attempts to find a match in the new frame in order to locate the target. One common problem is the low discriminative power of superpixel resulting in ambiguous matches. It then requires a

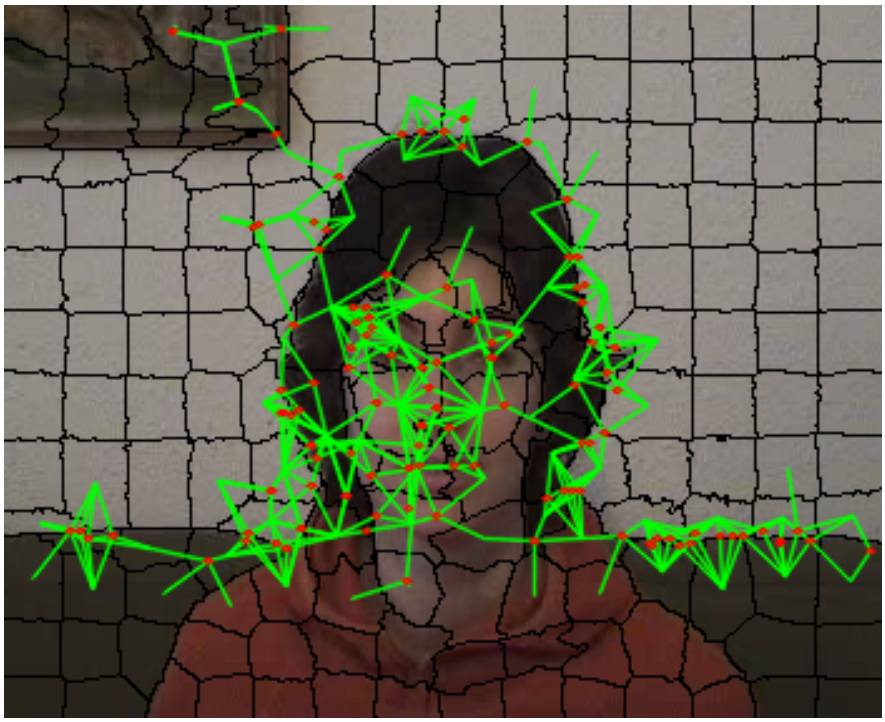


Figure 4.1 Decomposition of a frame into SPiKeS. Superpixels (black) structured by keypoints (red dot) linked by vectors (green).

complex strategy for matching.

Better features for matching are the keypoints. Because of their saliency and invariance to transformations, a keypoint-based appearance model can be matched efficiently even in case of occlusion and deformation. Nonetheless, keypoint-based trackers (Nebehay and Pflugfelder, 2014, 2015; Hare et al., 2012; Bouachir and Bilodeau, 2015) often fail to represent uniform regions, where no keypoints can be found.

Therefore, we hypothesize that superpixels and keypoints can complement each other. An object can always be segmented into superpixels but their lack of discriminative power makes them hard to match. Conversely, keypoints are more reliable to match but they poorly represent uniform-colored and non-textured regions. In our method, we propose to combine the assets of these two features in a single one : a Superpixel-Keypoints structure (SPiKeS). This is our first contribution. Figure 4.1 illustrates a frame decomposed into SPiKeS. Notice that keypoints contributing to a SPiKeS can be inside the superpixel or nearby. A single keypoint can contribute to many SPiKeS. Incomplete SPiKeS are possible if there is no keypoint around. In that case, they are only described with the superpixel.

Our second contribution is the design of a tracker that capitalizes on the SPiKeS. Experi-

mental results show that our SPiKeS-based tracker performs well in numerous challenging situations and performs favorably against state-of-the-art trackers.

The paper is structured as follows. Section 4.2 presents works related to ours, i.e. trackers based on superpixels or keypoints. Our combination of these two features to build a SPiKeS is described in Section 4.3. Then, section 4.4 shows how this new feature can be integrated into a tracking framework for robust target location estimation. Finally, the evaluation of section 4.5 compares the proposed tracker to the state-of-the-art.

4.2 Background

The idea of combining keypoints with other features for tracking has been exploited in (Yang et al., 2013). The RGB and LBP histograms are extracted from patches to create an appearance model. SIFT keypoints (Lowe, 2004) are then detected and their disposition is described by a circular histogram that represents a global geometric structure of the target. Our method is more flexible against deformation as each SPiKeS has its own keypoint structure, which allows local deformations.

Instead of patches, Liu et al. (Liu et al., 2011) proposed a tracker based on superpixels and SURF keypoints (Bay et al., 2006). But unlike our proposition, their matching step only involves keypoints. The superpixels are only used for their boundary evidence, which is a useful clue when updating the model. Indeed, a new keypoint belonging to the same superpixel than a matching keypoint tends to be a part of the target since every point within a superpixel is likely to belong to the same object. We also consider this benefit in our method but in addition, because we are matching superpixels, even if there is no matching keypoint inside them, new keypoints can still be added. Therefore, our model update is more accurate.

Our localization process is inspired by (Nebehay and Pflugfelder, 2014, 2015). Their approach assigns to each matching keypoint a vote for the center of the target, allowing keypoints to locate the target independently from each other. Hierarchical clustering then converges to a consensus of votes such that outliers are removed. Finally, the selected votes estimate the position as a simple center of mass. Furthermore, Bouachir et al. (Bouachir and Bilodeau, 2015) proposed to weight the votes according to the reliability of keypoints. We do the same, but instead of voting with keypoints, we vote with SPiKeS.

4.3 Superpixel-Keypoints Structure

As shown in figure 4.2, a Superpixel-Keypoints Structure (SPiKeS) consists in a superpixel and all the keypoints found in a region of radius R around that superpixel's center. It implies that keypoints can be inside and outside the superpixel. Each keypoint is linked to the superpixel's center by a vector with magnitude and orientation. Therefore, a SPiKeS is a superpixel that acquired a spatial structure of keypoints, making it more discriminative. A SPiKeS without any keypoints is simply a superpixel.

4.3.1 SPiKeS definition

Let s be a superpixel and $\mathbf{k}$ the set of keypoints around s, we write the associated SPiKeS denoted by S as

$$S = \{(s, \mathbf{k}) \mid \mathbf{k} = (k_1, ..., k_n, ..., k_N), ||\mathbf{x}_n^k - \mathbf{x}^s|| < R\} \tag{4.1}$$

with $\mathbf{x}^s$ and $\mathbf{x}_n^k$ the centers of superpixel s and keypoint n respectively. N is the total number of keypoints found in a description region of radius R centered on $\mathbf{x}^s$. Thereby, we define a descriptor for a SPiKeS as being $f = \{h, \mathbf{d}^k, \mathbf{e}\}$ with

- h : HSV histogram of s
- $\mathbf{d}^k = \{d_1^k, ..., d_n^k, ..., d_N^k\}$ with d_n^k the descriptor of k_n.
- $\mathbf{e} = \{\vec{e}_1, ..., \vec{e}_n, ..., \vec{e}_N\}$ with $\vec{e}_n = \mathbf{x}_n^k - \mathbf{x}^s$ the vector from the superpixel's center to k_n

4.3.2 SPiKeS comparison

In order to compare two SPiKeS, we propose a measure of similarity based on their descriptors. Let $z(S_i, S_j)$ be the similarity score between SPiKeS S_i and S_j. Because color information of superpixel and keypoints structure are available, the score is a contribution of two terms :

$$z(S_i, S_j) = \begin{cases} z_c + z_k & \text{if } d(h_i, h_j) < \theta_c \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

The first term z_c represents the similarity between superpixel's color histograms

$$z_c = \exp(-d(h_i, h_j)) \tag{4.3}$$

with $d(.,.)$ being the Bhattacharyya distance measure.

The second term z_k represents the similarity between keypoints structure. The higher the

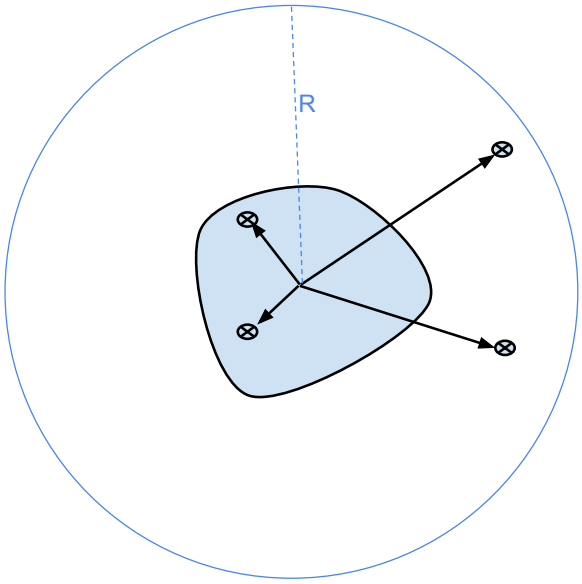


Figure 4.2 SPiKeS representation. Keypoints are found inside or nearby the superpixel in a region of radius R around its center. Keypoints relative positions are given by vectors.

number of matching keypoints between $\mathbf{k}_i$ and $\mathbf{k}_j$, the higher the score. Moreover, if both keypoints of a matching pair are positioned similarly with respect to their superpixel's center, the score should also increase. Thus we define

$$z_k = \sum_{k_m \in \mathbf{k}_i} \sum_{k_n \in \mathbf{k}_j} \gamma_{mn} p_{mn} \tag{4.4}$$

with $p_{mn} = 1$ if k_m and k_n match, else $p_{mn} = 0$. The factor γ_{mn} weights the contribution of a keypoints match by comparing edges e_m and e_n. We compute γ_{mn} with the vector difference magnitude normalized by the diameter of the description region :

$$\gamma_{mn} = \exp\left(-\frac{||\vec{e'_m} - \vec{e'_n}||}{2R}\right) \tag{4.5}$$

Note that to benefit from keypoints rotation invariance, e'_m and e'_n are the vectors e_m and e_n reoriented according to the principal orientations given by keypoints k_m and k_n respectively. Finally, the threshold θ_c ensures a minimum of color similarity to handle the case of wrong matching keypoints resulting in a high z_k.

4.4 The SPiKeS Tracker

In model-free tracking, the information about the target location in the first frame is given by a bounding box. The SPiKeS are extracted from it to represent the appearance model. In the subsequent frames, after oversegmentation and keypoint detection, we build SPiKeS and locate those who match the model. Then, each matching SPiKeS votes for a position in the frame. The target's center is estimated from all votes. If no occlusion is detected, the model is updated. These tracking steps are illustrated in figure 4.3.

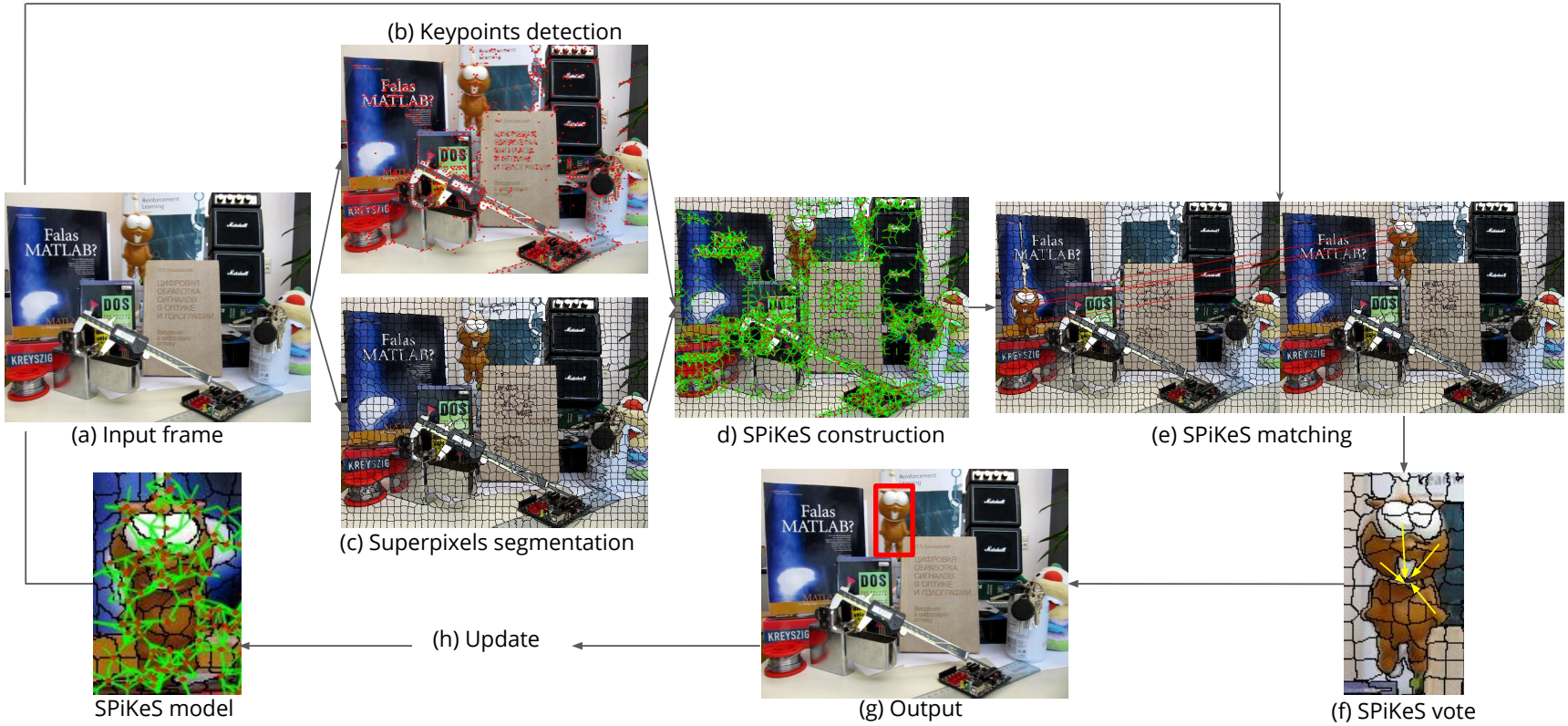


Figure 4.3 Tracking steps of our SPiKeS-based tracker. Keypoints detection (b) and superpixels segmentation (c) are processed on an input frame (a). Each superpixel forms a SPiKeS with its surrounding keypoints (d). Our SPiKeS model is matched with the new SPiKeS and the matching ones vote for the target's center (f) . The model is updated from the estimated bounding box (g) if no occlusion occurs.

4.4.1 Model

From the initial bounding box, we first detect keypoints, store them in a pool $\mathbf{K^f}$ and combine them with the N_m extracted superpixels to form our model of SPiKeS : $\mathbf{S}^m = \{S_1^m, ..., S_i^m, ..., S_{N_m}^m\}$. Then, SPiKeS is assigned a vote vector $\mathbf{v}_i$ such that it can locate independently the target's center :

$$\mathbf{v}_i = \mathbf{x}_0^T - \mathbf{x}_{S_i^m} \tag{4.6}$$

with $\mathbf{x}_0^T$ the target's center at time $t = 0$, known as the center of the initial bounding box. We refer to $\mathbf{x}_{S_i^m}$ as the center of S_i^m which is equivalent to its superpixel's center.

In addition, we extract keypoints in a surrounding region of the bounding box to keep a keypoint background model $\mathbf{K^b}$, which will help to detect occlusion similarly to Wang and Nevatia (2015).

4.4.2 Matching

During tracking, we extract a pool $\mathbf{S}^q = \{S_1^q, ..., S_j^q, ...S_{N_q}^q\}$ of N_q SPiKeS from the entire incoming frame. Afterwards, matching is executed in several steps. The first one is looking for the nearest neighbour $S_{j^*}^q \in \mathbf{S}^q$ of every $S_i^m \in \mathbf{S}^m$:

$$(S_i^m, S_{j^*}^q) = \underset{j}{\text{argmax}}(z(S_i^m, S_j^q)) \quad i = 1, ..., N_m \tag{4.7}$$

However, $S_{j^*}^q$ could be the nearest neighbour of different S_i^m meaning a many-to-one match. Since a one-to-one match is required, only the highest score is kept :

$$(S_{i^*}^m, S_{j^*}^q) = \underset{i}{\text{argmax}}(z(S_i^m, S_{j^*}^q)) \tag{4.8}$$

At this point, there are now $L \leq N_q$ one-to-one matches that we refer to as pairs of matches $M_l = (S_l^m, S_l^q)$ with $l = 1, 2, ...L$.

The next step consists in the rejection of wrong pairs of matches. Firstly, a given SPiKeS of the model may not have a valid match in a given new frame, e.g. when a part is not visible. In this case, the nearest neighbour has a low matching score relative to a threshold and it can be discarded. We set a different value for the threshold according to the presence or not of matching keypoints. Indeed, if there are no keypoint matches, only the color provides the match between SPiKeS. As we already set a color threshold θ_c, the minimum value of the matching score is $e^{-\theta_c}$ according to equation 4.2. On the other hand, if there are matching keypoints, the score will always be higher than this minimum value, thus the threshold is set

higher.

Secondly, as we assume the target motion is smooth and continuous in time, a match M_l is also considered inconsistent if the displacement between S_l^m and S_l^q is too large with respect to recent motion.

Formally, a matching pair M_l is valid and not discarded if and only if

$$z(S_l^m, S_l^q) > \begin{cases} e^{-\theta_c} & \text{if } N_{match}^{kp} = 0 \\ e^{-\theta_c} + \lambda_1 & \text{otherwise} \end{cases} \tag{4.9}$$

and

$$||\mathbf{x}_{S_l^m} - \mathbf{x}_{S_l^q}|| < ||\mathbf{x}_{t-1} - \mathbf{x}_{t-2}|| + \lambda_2 \tag{4.10}$$

with N_{match}^{kp} the total number of foreground keypoints matches, λ_1 a score threshold parameter and λ_2 a motion constraint parameter.

4.4.3 Location Estimation

Once the L^* retained matching pairs have been determined, each S_l^q votes for a position in the frame according to the vote vector given by its respective S_l^m :

$$\mathbf{x}_l = \mathbf{x}_{S_l^q} + \mathbf{v}_l \tag{4.11}$$

The estimated target location is computed by a weighted average of the votes :

$$\mathbf{x}_t^T = \frac{\sum_l^{L^*} \omega_l \phi_l \mathbf{x}_l}{\sum_l^{L^*} \omega_l \phi_l} \tag{4.12}$$

The factors ω_l and ϕ_l, as introduced in (Bouachir and Bilodeau, 2015), are the persistence and predictive factor of S_l^m. They give more importance to SPiKeS that often match and vote correctly for the target's center. More details are given in the next section.

4.4.4 Update

Subsection 4.4.1 introduced $\mathbf{K^f}$ and $\mathbf{K^b}$, our models of foreground and background keypoints. During the matching process, keypoints belonging to $\mathbf{K^b}$ are matched at the same time as the foreground ones. Once the new bounding box has been evaluated, if the number of keypoints inside it matching the background model exceeds a threshold θ_o, an occlusion is detected.

Therefore, no update takes place and the next frame is processed. Otherwise, if no occlusion occurs, the following update scheme is applied.

Step 1 : *Descriptors and votes update.* For each valid match (S_l^m, S_l^q), the SPiKeS descriptor f defined in subsection 4.3.1 is updated with :

$$f_l^m = (1 - \alpha_f) f_l^m + \alpha_f f_l^q \tag{4.13}$$

This simple formula adapts the model to gradual change of illumination by updating the color of the superpixels and the position and description of the keypoints.

For a non-rigid target, local parts tend to move with respect to the center. Vote vectors need to be modified according to the SPiKeS new position to take into account local deformations :

$$\mathbf{v}_l^m = (1 - \alpha_v)\mathbf{v}_l^m + \alpha_v(\mathbf{x}_t^T - \mathbf{x}_{S_l^q}) \tag{4.14}$$

As SPiKeS are the "parts" of our model, these terms are used interchangeably. A part that matches more often means that it is easier to identify and constitutes a stable part of the model. Consequently, this part should have more weight in the final vote because it has proved its reliability by the persistence of its matches. This persistence is interpreted as a factor ω. At the initialization, we consider every SPiKeS from the initial bounding box equally reliable and set an initial weight $\omega^0 = 1$ which is updated as

$$\omega_i^{t+1} = (1 - \beta)\omega_i^t + \beta \mathbb{1} \tag{4.15}$$

with $\mathbb{1} = 1$ if S_i is a matching SPiKeS, $\mathbb{1} = 0$ otherwise.

However, suppose an unexpected part of the background is included in our model. It could match as often as a foreground part if it is also present in the other frames. In that case, the persistence factors ω would be the same while the foreground part should have more importance. It can be observed on figure 4.4 that, as a background SPiKeS will not follow the target, the center estimated by its vote will be far from the predicted location, whereas the foreground SPiKeS votes will be closer. To leverage this behaviour, we introduce a predictive factor ϕ. Given a factor $\phi_l^0 = 1$ at time $t = 0$ for a SPiKeS l belonging to the model, the predictive factor ϕ_l is updated such that it increases if the local prediction $\mathbf{x}_l$ given by the vote is near the final location :

$$\phi_l^{t+1} = \phi_l^t + \exp(-||\mathbf{x}_l - \mathbf{x}_t^T||^2) \tag{4.16}$$

Those two factors ω and ϕ allow a SPiKeS to be more reliable if it often matches and votes correctly.

Step 2 : *SPiKeS insertion.* To handle appearance changes such as pose change showing parts not visible in the initial bounding box, one needs to add those new parts to the model. The main problem at this step is to avoid adding background parts that make the model drift. Therefore, instead of naively adding all the SPiKeS from the bounding box, we select only superpixels and keypoints that will make good SPiKeS candidates. Figure 4.5 illustrates how superpixels and keypoints help each other selecting good candidates. As the introduction stated, a keypoint inside a matching SPiKeS is assumed to belong to the target because of the boundary evidence given by the superpixel. Indeed, all the points inside that area are likely to belong to the same object. Similarly, a superpixel containing a matching foreground keypoint is more likely to belong to the target.

Let $\mathbf{s}_c$ and $\mathbf{k}_c$ be the sets of superpixels and keypoints candidates that meet these conditions. At first, the set $\mathbf{k}_c$ is added to the foreground keypoints pool $\mathbf{K^f}$. Then, the old SPiKeS from $\mathbf{S}^m$ refresh their structure with those new keypoints. Finally, we add the new SPiKeS made from the superpixels in $\mathbf{s}_c$ and the updated $\mathbf{K^f}$.

In order to complete the keypoint-based background model, keypoints detected around the estimated bounding box are added to $\mathbf{K^b}$ if they did not match the background keypoints.

Step 3 : *SPiKeS deletion.* As we add SPiKeS, our model grows and increases the complexity of the matching process. Furthermore, some SPiKeS may be irrelevant like redundant or background SPiKeS, which need to be deleted. To keep a reasonable number of SPiKeS in our model, once a maximum size N_m^{max} is exceeded, the $(N_m - N_m^{max})$ weakest SPiKeS are removed based on their persistence factor ω. The same discarding method apply for $\mathbf{K^f}$ and $\mathbf{K^b}$. Thus a persistence factor ω^k is assigned to each keypoint, updated similarly to equation 4.15, so that the weakest ones can be identified.

4.5 Experiments

In this section, we first present details of our implementation and the values for our parameters. Afterwards, we evaluate our method with the procedure proposed by (Wu et al., 2013) and compare our results to state-of-the-art trackers.

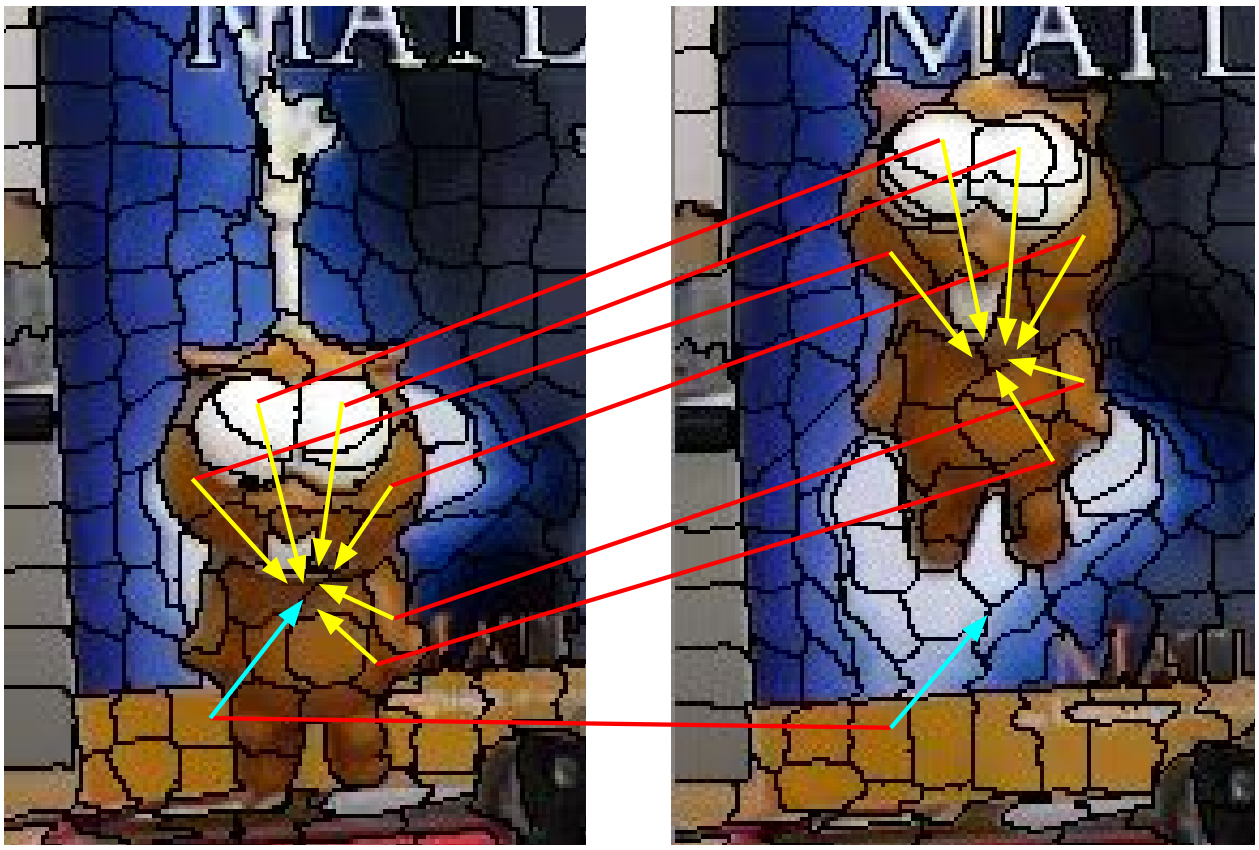


Figure 4.4 A wrong vote of a background SPiKeS included in the model (cyan) will have a weak predictive factor ϕ.

4.5.1 Experimental setup

For the oversegmentation, we choose the SEEDS superpixels (den Bergh et al., 2014) which have smooth boundaries and similar shapes in addition to be one of the fastest superpixel segmentation in the litterature. The size of a superpixel depends on the initial bounding box of dimension $w_B \times h_B$. In order to have about 30 superpixels included in the initial bounding box, a frame of dimension $w \times h$ should be segmented in $N = \frac{wh}{30 w_B h_B}$ superpixels of diameter $D^s = \sqrt{\frac{wh}{N}}$. Their HSV color histogram is quantified in $6 \times 6 \times 6$ bins and normalized. Similarly to (Hong et al., 2014), Grabcut (Rother et al., 2004) is used on the first frame to select foreground superpixels inside the given bounding box. This process makes the model more accurate as it avoids including background superpixels in the initial model. As for the keypoints and their descriptors, we use the SIFT algorithm (Lowe, 2004) which produces scale and rotation invariant keypoints robust against illumination variation. A match between keypoints is defined as proposed in (Lowe, 2004) with a ratio threshold $\theta_{lo} = 0.75$. When

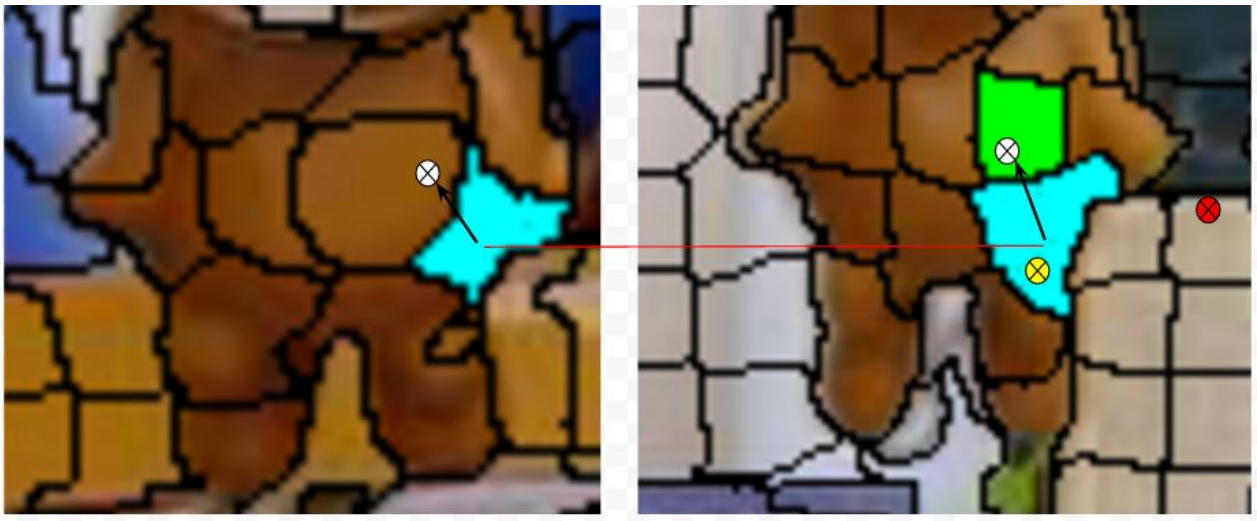


Figure 4.5 A new keypoint (yellow, right) inside a matching superpixel (cyan) can be added to $\mathbf{K^f}$ because this superpixel belongs to the target, unlike the red keypoint. In the same way, a new superpixel (green, right) can be added to $\mathbf{S}^m$ because it includes a matching keypoint (white).

building the SPiKeS, each superpixel searches its keypoints in a surrounding region of radius $R = 2D^s$. We limit the sizes of $\mathbf{S}^m$ to 3 times the initial size and $\mathbf{K^f}$ and $\mathbf{K^b}$ to 1000. During the matching process, the color threshold is set to $\theta_c = 0.75$, the score parameter to $\lambda_1 = 1$ and the motion constraint factor to $\lambda_2 = 4D^s$. At the update stage, the occlusion parameter is set to $\theta_o = 3$. A smooth appearance adaptation is obtained with interpolation factor $\alpha_v = \alpha_f = 0.1$ and learning factor $\beta = 0.1$. Finally, when a new SPiKeS is added, it starts with a weak persistence factor $\omega_{min} = 0.1$ such that it could be discarded quickly if it does not match directly in the next frame. Our results can be reproduced with our C++ implementation available online [2]. The following evaluation gives an average of 3 frames per second on a 3.4 GHz CPU with 8 GB memory, without code optimization. Note that most of the time is spent on superpixels segmentation, keypoints computation and matching, which are tasks that could be implemented on GPU to improve execution speed.

4.5.2 Evaluation

The CVPR2013 Online Object Tracking Benchmark (OTTB) of Wu et al.(Wu et al., 2013) allows us to evaluate our approach against 29 state-of-the-art trackers over a data set of 51 challenging sequences. The given groundtruth is a rectangular bounding box whose center

2. `https://github.com/fderue/SpikeS_T`

corresponds to the target location.

After running the one-pass evaluation (OPE), we obtain two types of graphs based on different metrics. The precision plot shows the percentage of frames for which the center location error (CLE) is lower than a Location Error Threshold (LET), with CLE computed as the Euclidian distance between the tracker estimated location and the groundtruth's center. On this plot, trackers are ranked by the precision obtained for LET = 20 pixels. The second graph is the success plot. It represents the percentage of frames for which the overlap ratio (OR) is larger than a given threshold. This ratio is computed between the intersection and union of the bounding box given by the tracker (B_T) and the groundtruth (B_G)

$$OR = \frac{S(B_T \cap B_G)}{S(B_T \cup B_G)} \tag{4.17}$$

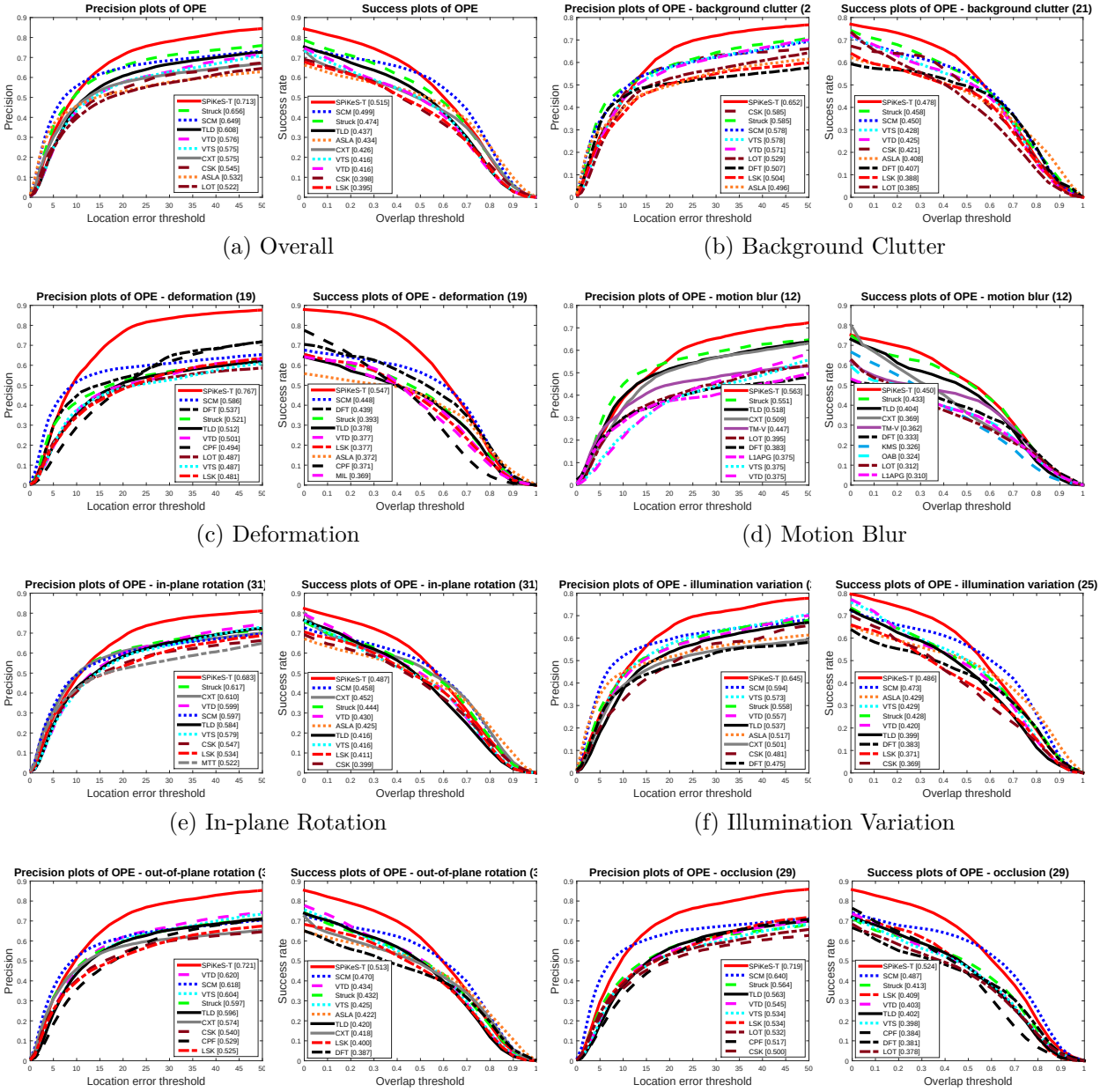
with S denoting the number of pixels of the covered surface. The ranking on this plot employs the area under the curve (AUC) value as it measures the overall performance instead of the success obtained for a single threshold.

Figure 4.6a shows the overall plots obtained from the whole dataset, while plots 4.6b-4.6h are obtained from subgroups gathering videos of the same challenging factor. Only the top ten are shown for clarity. In all of these cases, our tracker (SPiKeS-T) ranks first according to the criteria described above, proving that SPiKeS feature are robust tool against many challenging factors. Qualitative results for the top five trackers are presented on figure 4.9.

The overall precision plot shows that our tracker outperforms the others in term of center location, with a precision of 5.7% higher than the second one, Struck. As for the AUC value, it is only 1.6% higher than SCM which is actually a promising result as our tracker used only a fixed size bounding box. Even if our tracker locates correctly a target, in case of scale variation, the overlap with the groundtruth will definitely be lower. This is exactly the behavior we can see on the overall success plot, where our curve is higher than SCM for small overlap thresholds.

As Struck does not adapt to scale change either, we can fairly compare to it on the overall success and precision plots where our method reaches better performance on both. This is mainly due to our part-based model, which shows best results against deformation as seen in figure 4.6c. Indeed, our local parts, the SPiKeS, are very flexible as we do not enforce rigid connection between them. Each one is matched regardless of the others allowing large deformations. Moreover, a superpixel is a deformable part itself and better represents local deformation. This advantage has been exploited in our update scheme making our tracker more robust against background clutter than the other trackers as we observe on figure 4.6b.

For example, SCM updates its generative model with rectangular patches, which are less reliable than superpixels as patches can not adapt to the shape.



(a) Overall (b) Background Clutter

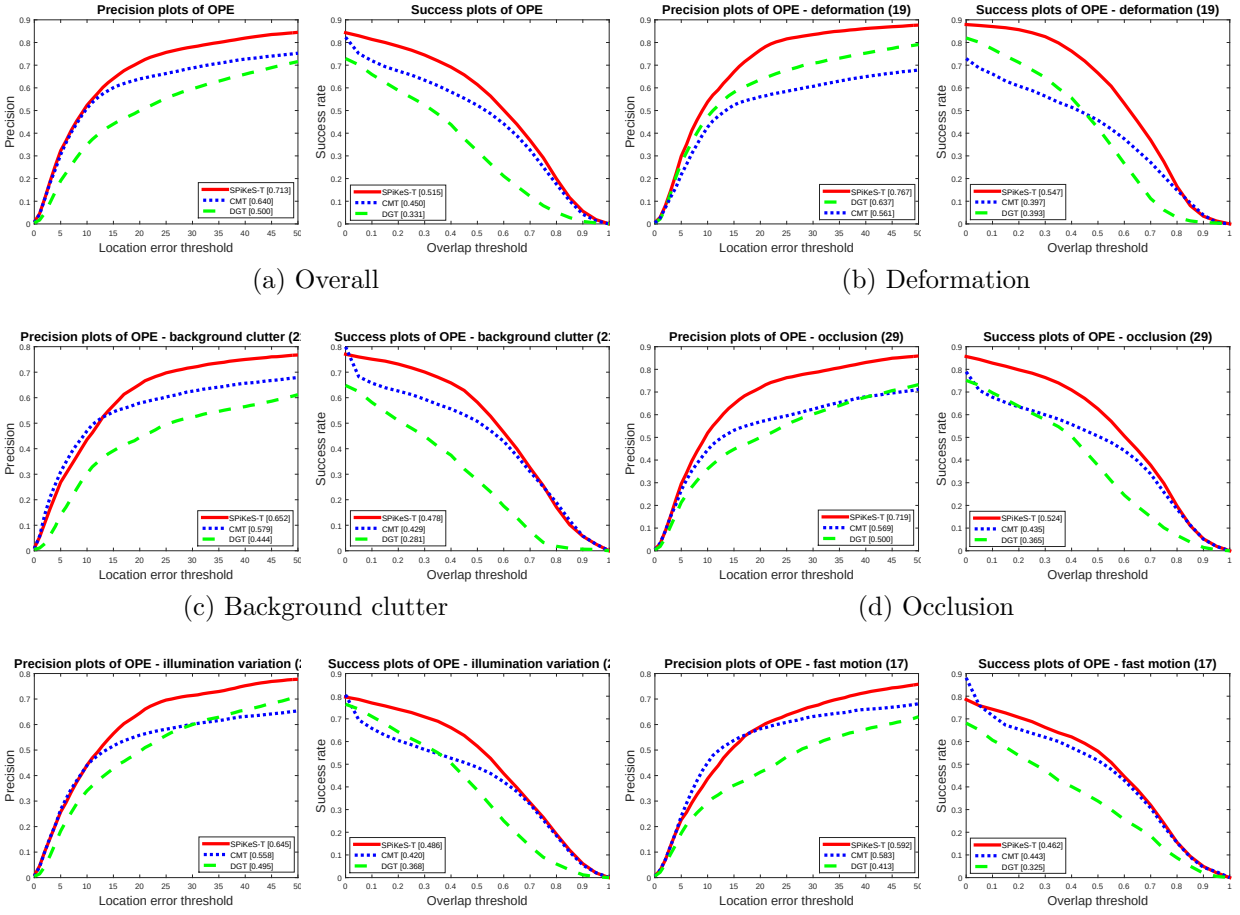
(c) Deformation (d) Motion Blur

(e) In-plane Rotation (f) Illumination Variation

(g) Out-of-plane Rotation (h) Occlusion

Figure 4.6 Precision and Success plots for the one-pass evaluation (OPE) on OTTB. The number into brackets is the number of videos in the subset.

As our goal is to show the benefits of the SPiKeS for tracking, we also compare our method to two specific recent trackers : CMT (Nebehay and Pflugfelder, 2015) and DGT (Cai et al., 2014). Both are part-based trackers which locate their target with votes like ours. However,



(a) Overall (b) Deformation

(c) Background clutter (d) Occlusion

(e) Illumination variation (f) Fast motion

Figure 4.7 Comparison of a superpixel-based tracker (DGT), a keypoint-based tracker (CMT) and ours (SPiKeS-T) on OTTB.

the former is a keypoint-only tracker while the latter is a superpixel-only tracker. As their codes are available online, we evaluate them on the benchmark of Wu et al. (Wu et al., 2013), keeping the default parameters given by the authors. Results in figure 4.7 show that our approach outperforms the other two, proving that combining superpixels and keypoints leads to a more robust tracking than using these features alone. More specifically, the results can be explained for different situations :

- *Deformation* : As they are all part-based trackers, they are more suited to handle deformations. To alleviate the lack of discriminativity of superpixels, DGT employs *spectral matching* to match a graph of superpixels. This technique requires the computation of an affinity matrix. However, for that matrix to be computationally manageable, constraints on the deformation must be set. Consequently, this tracker fails in case of heavy deformation. As for CMT, keypoints can be difficult to match when the tar-

get undergoes deformation, since some keypoints will disappear and new ones appear. Therefore, only a few matches will determine the location, which will be inaccurate if some of the matches are wrong. On the contrary, as an image can always be segmented in a same number of superpixels, numerous SPiKeS are candidates to be matched even in case of deformation. Moreover, since SpiKeS may have keypoints in common, a single keypoint can lead to several matches of SPiKeS resulting in a more accurate location.

- *Background clutter* : In this situation, the background distracts the tracking and often leads to a model drift. Where DGT will match wrong superpixels and CMT false keypoints, the more discriminative power of a SPiKeS helps in avoiding such ambiguous matches. Indeed, the color of a superpixel can avoid bad keypoint matches while the structure of keypoints can differentiate two superpixels of similar color. Compared to CMT, the boundary evidence brought by a superpixel avoids adding noisy keypoints to the model, as presented in figure 4.5 in the previous section. Furthermore, even if noisy SPiKeS are added to our model, the persistence and prediction factors favor reliable SPiKeS, which also prevents the model from drifting.
- *Occlusion* : On this curve, we see that DGT is less efficient. If the occluder has similar color as the target, it will be classified as foreground and no occlusion will be detected. Thus, it will not be able to avoid updating the model which will make it drift. To detect occlusion, it seems that keypoints are better but SpikeS has still an advantage over keypoints-only trackers. In case of a missed occlusion detection and an unwanted update, it is less probable to add bad keypoints to a SpiKeS thanks to the boundary of the superpixel. However, as keypoint-only trackers have no clue as whether a new keypoint belongs to the target, it is more likely that the model will drift due to a background keypoint added erroneously to the model.
- *Illumination variation* : In case of illumination variation, DGT tends to fail as it relies only on color and unlike CMT that detects BRISK keypoints (Leutenegger et al., 2011), our tracker uses SIFT keypoints which are designed to be robust against illumination variation.
- *Fast motion* : Constraints on the affinity matrix computed by DGT also limits the motion of each of its superpixels. This is why it performs poorly when there is fast motion. Our tracker adapts its motion constraint according to the target's motion.

It is also interesting to see the influence of other types of superpixels and keypoints to build the SPiKeS. Figure 4.8 compares different combinations of features such as SIFT (Lowe, 2004) and SURF (Bay et al., 2006) for keypoints and SLIC (Achanta et al., 2012) and SEEDS (den Bergh et al., 2014) for superpixels. Although the results are quite equivalent, the best

combination is not a surprise. SEEDS has shown to fit better to boundaries than SLIC in (den Bergh et al., 2014) and SIFT is more robust than SURF to illumination changes (Khan et al., 2011).

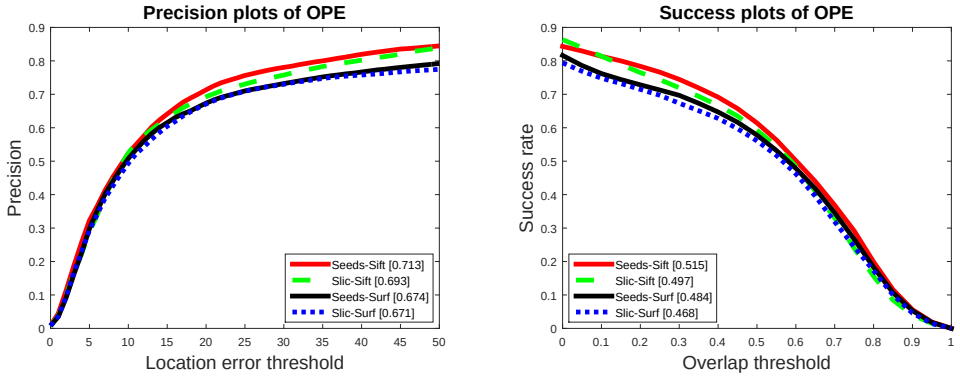


Figure 4.8 Influence of different superpixels-keypoints combinations on the overall performance on OTTB.

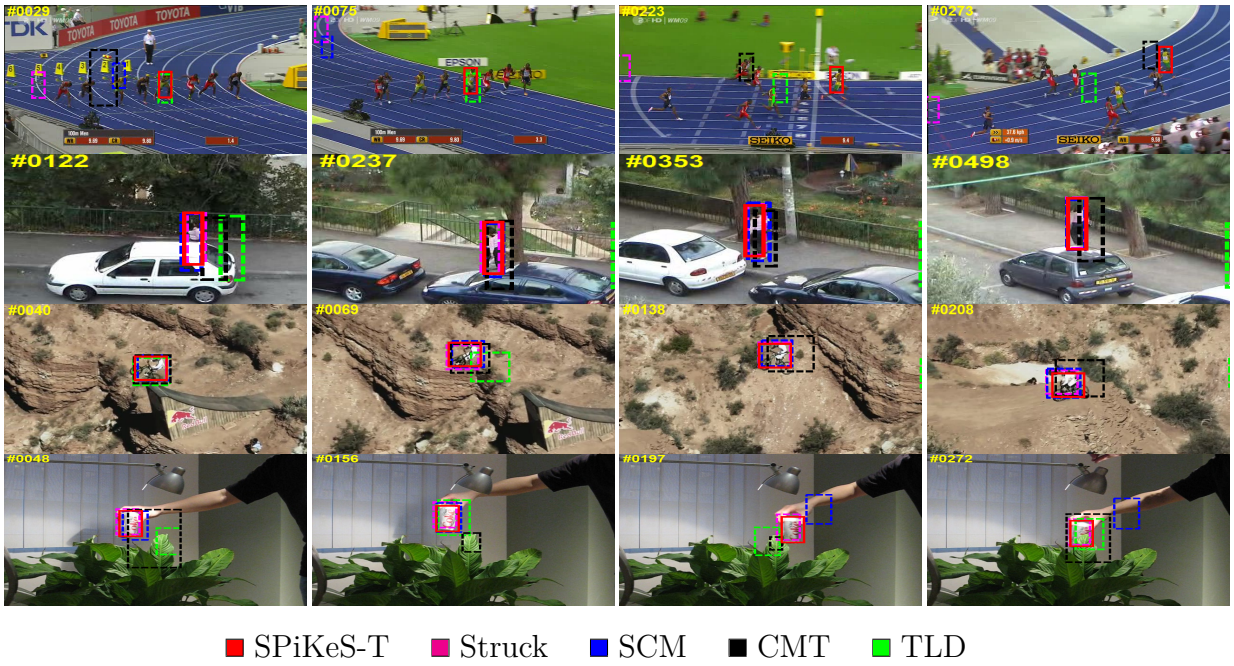


Figure 4.9 Qualitative results of top five trackers for sequences *bolt*, *woman*, *mountainBike*, *coke* from top to bottom.

4.6 Conclusion

In this paper, we proposed a novel feature combining superpixels and keypoints that we called SPiKeS. We showed that this new feature can be matched efficiently by a simple nearest neighbor technique. Therefore, we developed a SPiKeS-based tracker that leverages this matching to locate accurately target parts in a new frame. Furthermore, based on the SPiKeS properties, we provided a reliable update scheme that avoids the model to drift. Finally, the evaluation against the state-of-the-art shows promising results, as we outperform the Struck robust tracker, despite that our tracker does not yet include an adaptation to scale variation. In addition, our superior performance compared to superpixels-only and keypoints-only trackers demonstrates the benefits of combining these two features for more robust tracking. As a final word, we point out that the use of SPiKeS could advantageously be extended to other applications such as object detection and foreground segmentation.

RÉFÉRENCES

R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, et S. Susstrunk, “SLIC superpixels compared to state-of-the-art superpixel methods”, *TPAMI*, vol. 34, no. 11, pp. 2274–2282, 2012.

A. Adam, E. Rivlin, et I. Shimshoni, “Robust fragments-based tracking using the integral histogram”, dans *CVPR*, vol. 1, 2006, pp. 798–805.

S. Avidan, “Support Vector Tracking”, *PAMI*, vol. 26, no. 8, pp. 1064–1072, 2004.

B. Babenko, M. Yang, et S. Belongie, “Visual Tracking with Online Multiple Instance Learning”, *PAMI*, vol. 33, no. 8, pp. 1619–1632, 2011.

Q. Bai, Z. Wu, S. Sclaroff, M. Betke, et C. Monnier, “Randomized Ensemble Tracking”, dans *ICCV*, vol. 1, 2013, pp. 2040–2047.

H. Bay, T. Tuytelaars, et L. V. Gool, “Surf : Speeded up robust features”, dans *ECCV*, 2006.

W. Bouachir et G.-A. Bilodeau, “Part-based tracking via salient collaborating features”, dans *WACV*, 2015.

Z. Cai, L. Wen, Z. Lei, N. Vasconcelos, et S. Z. Li, “Robust Deformable and Occluded Object Tracking With Dynamic Graph”, *IEEE Transaction on Image Processing*, vol. 23, no. 12, pp. 5497–5509, 2014.

M. Danelljan, F. Khan, M. Felsberg, et J. van de Weijer, “Adaptive color attributes for real-time visual tracking”, dans *CVPR*, 2014.

M. V. den Bergh, X. Boix, G. Roig, et L. V. Gool, “SEEDS : Superpixels Extracted via Energy-Driven Sampling”, *IJCV*, 2014.

S. Hare, A. Saffari, et P. H. Torr, “Structured output tracking with kernels”, dans *ICCV*, 2011, pp. 263–270.

——, “Efficient online structured output learning for keypoint-based object tracking”, dans *CVPR*, 2012, pp. 1894–1901.

S. He, Q.-X.Yang, R. Lau, J. Wang, et M.-H. Yang, “Visual tracking via adaptive structural local sparse appearance model”, dans *CVPR*, 2013.

J. Henriques, R. Caseiro, P. Martins, et J. Batista, “Exploiting the circulant structure of tracking-by-detection with kernels”, dans *ECCV*, 2012.

——, “High-speed tracking with kernelized correlation filters”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.

Z. Hong, C. Wang, X. Mei, D. Prokhorov, et D. Tao, “Tracking using multilevel quantizations”, dans *ECCV*, 2014.

X. Jia, H. Lu, et M.-H. Yang, “Visual tracking via adaptive structural local sparse appearance model”, dans *CVPR*, 2012, pp. 1822–1829.

N. Y. Khan, B. McCane, et G. Wyvill, “Sift and surf performance evaluation against various image deformations on benchmark dataset”, dans *International Conference on Digital Image Computing : Techniques and Applications*, 2011, pp. 501–506.

J. Kwon et K. M. Lee, “Visual tracking decomposition”, dans *CVPR*, 2010.

S. Leutenegger, M. Chli, et R. Siegwart, “BRISK : Binary Robust Invariant Scalable Keypoints”, dans *ICCV*, 2011, pp. 2548–2555.

Y. Li, J. Zhu, et S. C. Hoi, “Reliable Patch Trackers : Robust Visual Tracking by Exploiting Reliable Patches”, dans *CVPR*, 2015, pp. 353–361.

Y. Liu, W. Zhou, H. Yin, et N. Yu, “Tracking Based on SURF and Superpixel”, dans *Sixth International Conference on Image and Graphics*, 2011, pp. 714–719.

D. G. Lowe, “Distinctive image features from scale-invariant keypoints”, *IJCV*, vol. 60, no. 2, pp. 91–110, 2004.

X. Mei et H. Ling, “Robust visual tracking using l1 minimization”, dans *ICCV*, 2009.

G. Nebehay et R. Pflugfelder, “Consensus-based matching and tracking of keypoints for object tracking”, dans *WACV*, 2014.

——, “Clustering of Static-Adaptive Correspondences for Deformable Object Tracking”, dans *CVPR*, 2015.

C. Rother, V. Kolmogorov, et A. Blake, “Grabcut : Interactive foreground extraction using iterated graph cuts”, *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 309–314, 2004.

J. Wang et Y. Yagi, “Many-to-Many Superpixel Matching for Robust Tracking”, *IEEE Transaction on Cybernetics*, vol. 44, no. 7, pp. 1237–1248, 2014.

S. Wang, H. Lu, F. Yang, et M.-H. Yang, “Robust Superpixel Tracking”, *IEEE Trans. Image Process.*, vol. 23, no. 4, pp. 1639–1651, 2014.

W. Wang et R. Nevatia, “Multistore tracker (muster) : A cognitive psychology inspired approach to object tracking”, dans *CVPR*, 2015, pp. 749–758.

Y. Wu, J. Lim, et M.-H. Yang, “Online Object Tracking : A Benchmark”, dans *CVPR*, 2013.

F. Yang, H. Lu, et M.-H. Yang, “Learning structured visual dictionary for object tracking”, *Image and Vision Computing*, vol. 31, no. 12, pp. 992–999, 2013.

W. Zhong, H. Lu, et M.-H. Yang, “Robust object tracking via sparsity-based collaborative model”, dans *CVPR*, 2012, pp. 1838–1845.

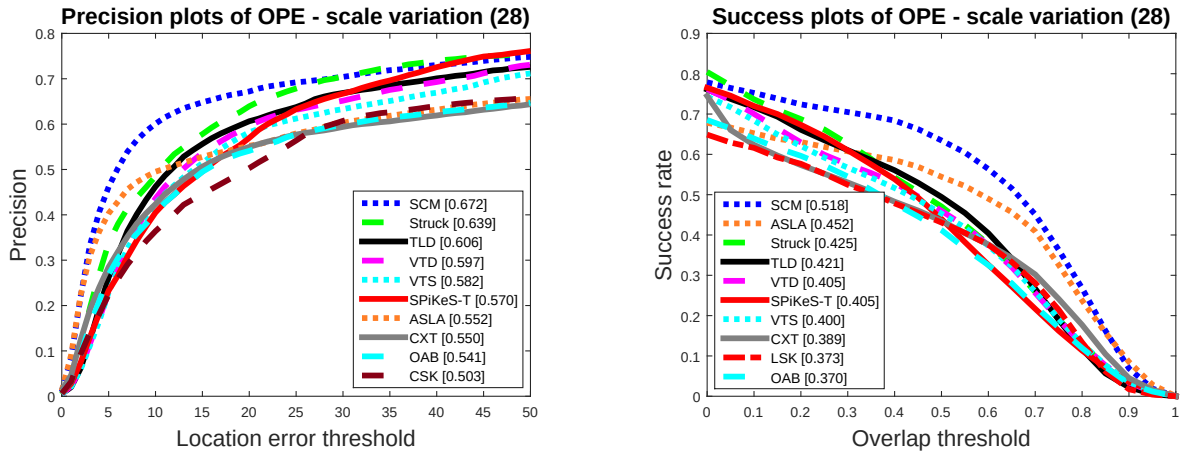
Q. Zhou, H. Lu, et M.-H. Yang, “Online multiple support instance tracking”, dans *Proc. IEEE Conf. Automatic Face and Gesture Recognition*, 2011, pp. 545–552.

CHAPITRE 5 DISCUSSION GÉNÉRALE

Ce chapitre est dédié aux limitations de l'approche proposée dans notre article. Différentes solutions sont discutées et pourront faire l'objet de travaux futurs.

Tel qu'énoncé dans l'article, la "mise à l'échelle" est une composante manquante à notre algorithme de suivi. Elle permet d'ajuster le rectangle englobant à la taille de la cible si cette dernière s'éloigne ou se rapproche de la caméra par exemple. La précision de la mise à l'échelle est évaluée par le critère *OR* ("overlap ratio" en anglais) qui calcule l'intersection sur l'union de la surface de la boite englobante estimée avec celle annotée (équation 4.17). Bien que les résultats globaux par rapport à ce critère semblent prometteurs, cela est principalement dû au fait que la taille des cibles ne changent pas beaucoup dans la majorité des séquences vidéo. Par conséquent, si la localisation est suffisament précise, une boite englobante de taille fixe reste une bonne approximation. On observe sur la figure 5.2a une localisation précise du centre mais la taille de la boite englobante est restée à ses dimensions initiales. Elle ne couvre alors pas toute la cible et le critère *OR* en sera affecté.

Parmi les séquences vidéo du banc de test de Wu et al. (2013), si l'on se penche sur celles où la principale difficulté est le changement d'échelle, il n'est alors pas surprenant que notre algorithme de suivi ne se situe que 6ème dans le classement (voir figure 5.1), tant pour la courbe de succès (fig. 5.1b) que celle de précision (fig. 5.1a). Pour rappel, cette dernière évalue évalue la précision de la localisation du centre (critère *CLE* dans l'article). La précision est affectée car une bonne localisation au fil de la séquence repose sur une mise à jour robuste. Puisque cette étape requiert l'information extraite de la boite englobante, elle dépend alors de la mise à l'échelle. Si l'on parvenait à adapter les dimensions de cette boite, les résultats face au facteur de mise à l'échelle seraient meilleurs et donc les résultats globaux aussi. Cependant, une mise à l'échelle imprécise pourrait affecter le reste du suivi. Dans notre méthode, une boite englobante trop grande par rapport à la cible incluera des éléments d'arrière-plan. Par conséquent, notre système va détecter une fausse occultation et ne pas mettre à jour le modèle, alors qu'il aurait dû. De même, si la boite est estimée trop petite, la mise à jour omettra des éléments importants car seulement ce qui est à l'intérieur de la boite est considéré comme nouveau candidat au modèle. Ainsi, une mauvaise estimation d'échelle pourrait dégrader le suivi alors qu'une boite englobante de taille fixe aurait été plus efficace. Cet inconvénient a été observé à travers différentes stratégies testées que nous présentons ici.

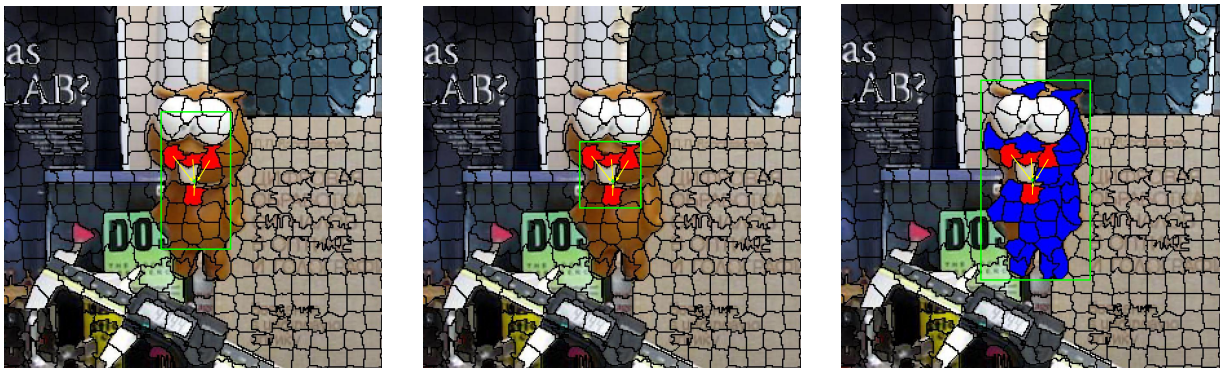


(a) Précision (CLE) (b) Succès (OR)

Figure 5.1 Résultats de SPiKeS-T face au changement d'échelle.

Mise à l'échelle basée sur les superpixels

Intuitivement, il suffirait d'ajuster la taille du rectangle afin qu'il englobe le plus de superpixels qui ont été mis en correspondance dans la nouvelle trame. Cependant, imaginons que les superpixels du modèle n'aient pas tous trouvé d'appariement. Par conséquent, la boite aura tendance à rétrécir (fig. 5.2b). Une solution serait de déterminer les superpixels pouvant potentiellement appartenir à la cible mais qui n'auraient pas été appariés. Similairement à Cai et al. (2014), nous avons testé un classifieur SVM entrainé sur la couleur des superpixels du modèle afin d'obtenir une classification grossière de la cible par rapport à l'arrière-plan. Puisque nous possédons déjà la localisation grâce aux votes, il suffit de tester différentes tailles de boite englobante centrée en cette position. La boite appropriée englobera le plus de superpixels appariés et de superpixels classés comme avant-plan par le classifieur, tout en rejetant un maximum de superpixels d'arrière-plan. Cette idée est illustrée sur la figure 5.2, où les superpixels appariés qui ont votés pour le centre sont rouges et ceux qui ont été classifiés positifs, comme avant-plan, sont bleus. Expérimentalement, certaines séquences vidéo donnent des résultats convaincants comme la voiture de la figure 5.3. Par contre, un arrière-plan de couleur similaire à la cible entrainera une mauvaise classification ainsi qu'une estimation d'échelle incorrecte. C'est la situation montrée sur la figure 5.4, où l'on voit que le changement d'illumination classifie trop de superpixels d'avant-plan (la lumière est aussi blanche que la robe de la chanteuse). Le rectangle englobant est alors agrandi à tort.



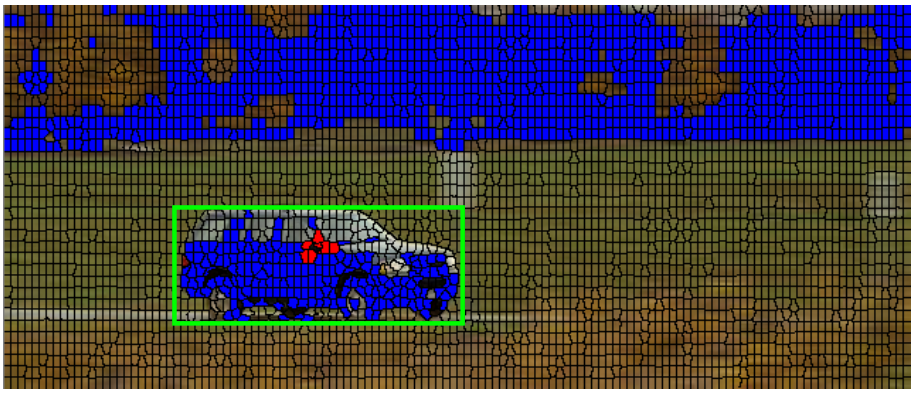
(a) Pas de mise à l'échelle, mêmes dimensions qu'à l'initialisation.

(b) Englober le plus de superpixels appariés : incorrect dû au manque d'appariements.

(c) Estimation correcte : englobe le plus de superpixels appariés et classifiés comme avant-plan (bleu).

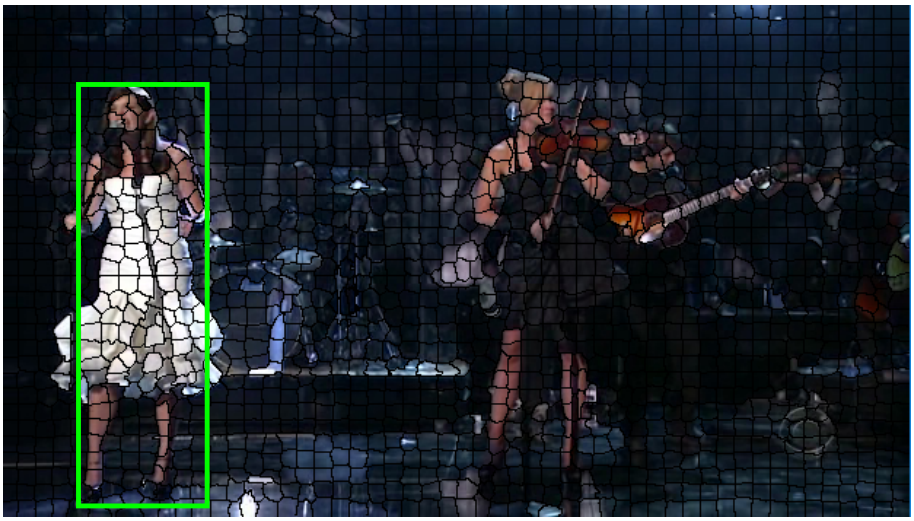
Figure 5.2 Centre de la cible correctement localisé par quelques superpixels appariés (rouge) et différentes estimations de l'échelle.

(a) Cible initiale, peu de superpixels composent la cible.

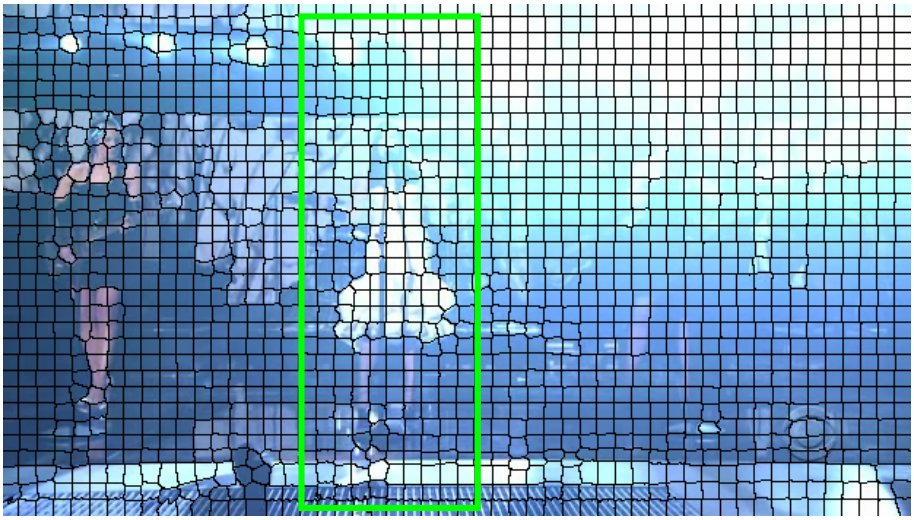


(b) La classification par couleur permet de détecter des superpixels d'avant-plan (bleu) autres que les superpixels appariés (rouge).

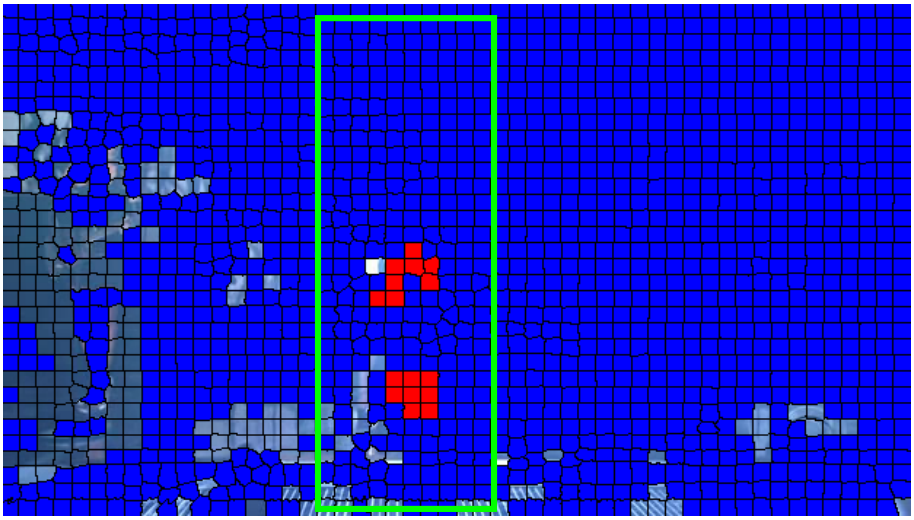
Figure 5.3 Mise à l'échelle ajustée grâce à la classification de superpixels d'avant-plan.



(a) Cible initiale, la majorité des superpixels sont blancs.



(b) Un changement d'illumination rend l'arrière-plan de la même couleur que la cible.



(c) Trop de superpixels classifiés comme avant-plan (bleu)

Figure 5.4 La mise à l'échelle échoue due à une classification incorrecte.

Mise à l'échelle basée sur les points clés

L'information d'échelle donnée par chaque point clé pourrait être utilisée pour estimer la variation d'échelle d'une manière similaire à l'algorithme de suivi de Bouachir and Bilodeau (2015). Cependant, Nebehay and Pflugfelder (2015) soulèvent que l'échelle détectée par les points clés n'est pas assez fiable. Ils considèrent alors plutôt les relations géométriques entre des paires d'appariements. L'idée est que si deux points clés se sont rapprochés, alors la taille de la cible a diminué tandis que s'ils se sont éloignés l'un de l'autre, la cible s'est agrandie. Mathématiquement, ce comportement est exprimé de la façon suivante. Soit un appariement i d'un point clé du modèle initial k_i^0 avec un point clé de la trame courante k_i^t, la taille du rectangle englobant S au temps t est calculé comme

$$S^t = \text{med}\left\{\frac{||x_i^t - x_j^t||^2}{||x_i^0 - x_j^0||^2}, i \neq j\right\} S^0 \tag{5.1}$$

où x représente la position du point clé k, S^0 la taille initiale du rectangle, med la médiane et i et j les indices allant de 1 au nombre total d'appariements.

En testant cette méthode avec notre algorithme, certaines séquences vidéos comme la chanteuse (fig. 5.5) fonctionnent très bien. Malgré un changement d'illumination, une grande quantité de points clés du modèle initial est toujours reconnaissable et permet d'évaluer la variation d'échelle. Cependant, puisque seul le modèle de points clés initial est utilisé, lorsqu'aucun de ces points clés ne peut être mis en correspondance, cette stratégie fait défaut. En outre, dans le cas d'un faible nombre de correspondances dont la plupart sont erronées, la mise à l'échelle sera aussi erronée. Une alternative que nous proposons est d'utiliser les points clés rajoutés au modèle lors de la mise à jour. Cependant, puisque ceux-ci peuvent être détectés à différentes échelles, il faudrait conserver l'échelle à laquelle ils ont été détectés. La formule (eq. 5.1) deviendrait alors :

$$S^t = \text{med}\left\{\frac{||x_i^t - x_j^t||^2}{||x_i^l - x_j^l||^2} S^l, i \neq j\right\} \tag{5.2}$$

avec $l < t$. Cette solution a également son désavantage. Supposons que l'échelle détectée des points clés au temps l soit mauvaise, cette erreur se propagera dans cette équation et toutes les futures mises à l'échelle en seront affectées. Même si, contrairement à une moyenne, la médiane n'est pas affectée par des données aberrantes (*outliers*), si les seuls points clés qui ont été mis en correspondance dans la nouvelle trame étaient ceux détectés à la mauvaise échelle, la médiane sera erronée aussi. Alors qu'en se fiant seulement au modèle initial, on est sûr de la taille de la boite englobante puisqu'elle est fournie à l'initialisation.

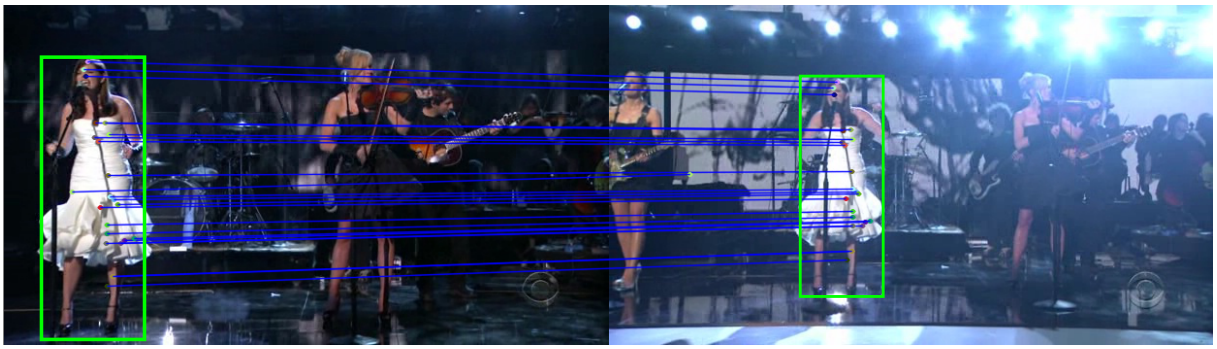


Figure 5.5 Mise à l'échelle par paires d'appariement de points clés.

La mise à l'échelle est un problème complexe. D'ailleurs, des articles entiers se focalisent seulement à améliorer des algorithmes de suivi existants en y apportant une mise à l'échelle. Citons notamment la méthode de Danelljan et al. (2014b) qui reprend celle de Henriques et al. (2015) et lui ajoute une mise à l'échelle à partir d'une localisation donnée. Il serait envisageable d'incorporer cette stratégie au sein de notre algorithme. Néanmoins, elle requiert le calcul d'autres caractéristiques, ce qui complexifierait davantage notre méthode.

En outre, des cas d'échec inhérents à notre méthode sont les suivants :

- S'il n'y a pas de points clés, l'identification des SPiKeS repose seulement sur la couleur, ce qui donnera aussi bien une bonne localisation si leurs couleurs sont différentes qu'une localisation imprécise dans le cas contraire. Une solution serait d'emprunter la technique de *spectral matching* utilisée par Cai et al. (2014) dans ce genre de situation mais cela surchargerait notre méthode.
- Si aucun SPiKeS ne trouve de correspondance, notre AS reste sur la position précédente et aucune mise à jour n'a lieu. A cause de la contrainte de mouvement, il est possible que même si l'on retrouve des correspondances X trames plus loin, elle ne soient pas acceptées. La cible est donc perdue. Comme expliqué dans l'article, la contrainte de mouvement est nécessaire dans le cas de distracteurs (voir fig. 1.3g) où les correspondances ambigues sont éliminées car leur mouvement n'est pas cohérent avec celui de la cible.

La figure 5.6 illustre un cas typique de situation complexe où notre algorihtme ne parvient pas à suivre la cible. Le rectangle englobant initial donne très peu d'information, le casque est de couleur uniforme et faiblement texturé, ce qui ne donne quasiment pas de points clés. Entre la trame 18 et 19, on observe un changement instantané d'illumination. Notre AS n'a pas le temps de s'adapter et considèrera les couleurs trop différentes. Par conséquent, aucune mise en correspondance n'a lieu et notre algorithme reste à la même position. Ironman a le temps de bouger d'une grande amplitude et changer d'apparence sans qu'on puisse mettre sa

position à jour. D'ailleurs, à peine 6 trames plus loin, l'apparence d'Ironman a complètement changée mais notre AS est resté sur l'ancienne apparence car il n'y a eu aucune mise à jour. Un autre exemple est le passage de la trame 114 à 115 où les apparences sont totalement différentes et le déplacement de la cible très rapide.

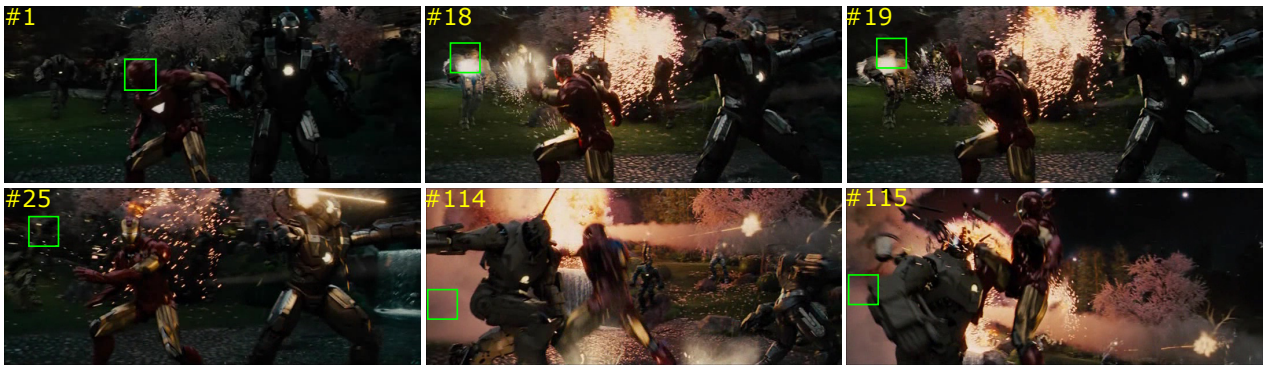


Figure 5.6 Séquence complexe *Ironman* où notre algorithme échoue à suivre sa cible.

Enfin, bien que la complémentarité entre superpixel et point clé est avantageuse comme nous l'avons montré au cours de ce projet, un inconvénient est le calcul de ces deux caractéristiques. En effet, les opérations les plus demandantes en terme de calcul sont la segmentation en superpixels, la détection et la description des points clés. L'algorithme "glouton", utilisé pour la mise en correspondance, est également lourd à calculer de par sa complexité en $\mathcal{O}(n^2)$ avec n le nombre de SPiKeS. Cependant, il est plus rapide que l'algorithme "hongrois" (Kuhn, 1955), de complexité $\mathcal{O}(n^3)$, et plus efficace pour notre méthode. En effet, il s'est révélé expérimentalement qu'obtenir un optimum global donnait davantage de mauvais appariements que de prendre itérativement les appariements de similarité maximale tel que nous le faisons.

Sur le banc de test de Wu et al. (2013), le temps de calcul moyen est de 3 trames par seconde (TPS). Bien que le but principal de cette recherche ne soit pas axé sur un critère de vitesse, nous nous sommes intéressés à exploiter les capacités graphiques des processeurs de type GPU dans une optique d'accélération de notre algorithme de suivi. En parallélisant la segmentation en superpixels sur GPU, nous pouvons atteindre une exécution 3 fois plus rapide, soit une moyenne de 9 TPS. En réalité, la segmentation elle-même est 10 fois plus rapide. Cependant, puisque l'algorithme au complet n'a pas été implémenté sur GPU, il faut prendre en compte les temps de transfert CPU-GPU qui réduisent inévitablement les performances globales. Pour avoir un réel gain en performance, il faudrait que les seuls transferts à considérer soient les images et le résultat de l'algorithme, soit la localisation de la cible. Une implémentation complète sur GPU pourrait être envisageable comme travail futur. À titre de comparaison, les algorithmes de suivi basés sur seulement des superpixels comme DGT (Cai et al., 2014)

ou points clés comme CMT (Nebehay and Pflugfelder, 2015), évalués sur le même ordinateur que pour nos expériences, atteignent une moyenne de 13 TPS. Par contre, un AS à filtre de corrélation comme CSK (Henriques et al., 2012) peut atteindre une centaine de trames par seconde alors qu'un AS très précis comme MDNet (Nam and Han, 2015) reposant sur des CNN s'exécute à seulement 1 TPS, même avec l'utilisation du GPU.

CHAPITRE 6 CONCLUSION ET RECOMMANDATIONS

L'objectif de ce travail était de développer une nouvelle méthode de suivi visuel d'objet, sans modèle a priori, à travers une séquence vidéo. Nous avons vu que ce problème pouvait se révéler complexe dû à de nombreuses situations difficiles dans lesquelles la cible pouvait se trouver. Les déformations de la cible, son occultation ou encore les changements d'illumination en sont des cas typiques. La revue de la littérature a présenté différentes stratégies développées à ce jour pour pallier à ces difficultés, certaines étant plus appropriées que d'autres pour résoudre des situations spécifiques. Dans notre cas, nous avons choisi un modèle par parties qui a l'avantage d'être naturellement robustes aux déformations et aux occultations.

De par leur succès à s'adapter aux frontières, les superpixels ont été choisis pour constituer les parties locales du modèle d'apparence. Grâce à leur mise en correspondance dans une nouvelle trame, chacun de ces superpixels appariés est capable de localiser la cible individuellement en votant pour le centre de la cible. Ainsi, en cas d'occultation, la cible peut être localisée précisément même si seulement une fraction du modèle est visible. L'inconvénient des superpixels est leur manque de discriminativité s'ils sont décrits uniquement par leur couleur. Par conséquent, leur mise en correspondance est souvent ambigue. Pour surmonter ce problème tout en gardant une stratégie d'appariement simple, nous avons amélioré le superpixel avec des points clés afin de le rendre plus distinctif. Ce nouvel élément, appelé *SPiKeS*, constitue notre première contribution. La seconde est de l'intégrer à une procédure de suivi qui met à profit les avantages des superpixels et des points clés. L'algorithme de suivi obtenu a montré des performances très convaincantes par rapport aux méthodes proposées dans l'état de l'art. Plus spécifiquement, nous avons pu démontrer l'avantage de la combinaison superpixel-points clés en obtenant des résultats supérieurs à des approches récentes se fiant uniquement soit qu'à des superpixels, soit qu'à des points clés. Néanmoins, nous avons soulevé plusieurs limitations, dont celle d'une mise à l'échelle manquante qui pourra faire l'objet de travaux futurs.

RÉFÉRENCES

R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, et S. Susstrunk, “SLIC superpixels compared to state-of-the-art superpixel methods”, *TPAMI*, vol. 34, no. 11, pp. 2274–2282, 2012.

A. Adam, E. Rivlin, et I. Shimshoni, “Robust fragments-based tracking using the integral histogram”, dans *CVPR*, vol. 1, 2006, pp. 798–805.

S. Avidan, “Support Vector Tracking”, *PAMI*, vol. 26, no. 8, pp. 1064–1072, 2004.

B. Babenko, M. Yang, et S. Belongie, “Visual Tracking with Online Multiple Instance Learning”, *PAMI*, vol. 33, no. 8, pp. 1619–1632, 2011.

Q. Bai, Z. Wu, S. Sclaroff, M. Betke, et C. Monnier, “Randomized Ensemble Tracking”, dans *ICCV*, vol. 1, 2013, pp. 2040–2047.

H. Bay, T. Tuytelaars, et L. V. Gool, “Surf : Speeded up robust features”, dans *ECCV*, 2006.

D. Bolme, J. Beveridge, B. Draper, et Y. Lui, “Visual object tracking using adaptive correlation filters”, dans *CVPR*, 2010.

W. Bouachir et G.-A. Bilodeau, “Part-based tracking via salient collaborating features”, dans *WACV*, 2015.

G. Bradski, “Computer Vision Face Tracking as a Component of a Perceptual User Interface”, dans *IEEE Workshop Applications of Computer Vision*, 1998, pp. 214–219.

Z. Cai, L. Wen, Z. Lei, N. Vasconcelos, et S. Z. Li, “Robust Deformable and Occluded Object Tracking With Dynamic Graph”, *IEEE Transaction on Image Processing*, vol. 23, no. 12, pp. 5497–5509, 2014.

D. Comaniciu, V. Ramesh, et P. Meer, “Kernel-Based Object Tracking”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003.

M. Danelljan, F. Khan, M. Felsberg, et G. Hager, “Accurate scale estimation for robust visual tracking”, dans *BMVC*, 2014.

M. Danelljan, F. Khan, M. Felsberg, et J. van de Weijer, “Adaptive color attributes for real-time visual tracking”, dans *CVPR*, 2014.

M. V. den Bergh, X. Boix, G. Roig, et L. V. Gool, “SEEDS : Superpixels Extracted via Energy-Driven Sampling”, *IJCV*, 2014.

P. Felzenszwalb, R. Girshick, D. McAllester, et D. Ramanan, “Object detection with discriminatively trained part-based models”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

K. Fukunaga, “Introduction to statistical pattern recognitio”, dans *Tools and Algorithms for the Construction and Analysis of Systems*. Boston : Academic Press, 1990.

R. Girshick, J. Donahue, T. Darrell, et J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation”, dans *CVPR*, 2012.

S. Hare, A. Saffari, et P. H. Torr, “Structured output tracking with kernels”, dans *ICCV*, 2011, pp. 263–270.

——, “Efficient online structured output learning for keypoint-based object tracking”, dans *CVPR*, 2012, pp. 1894–1901.

S. He, Q.-X.Yang, R. Lau, J. Wang, et M.-H. Yang, “Visual tracking via adaptive structural local sparse appearance model”, dans *CVPR*, 2013.

J. Henriques, R. Caseiro, P. Martins, et J. Batista, “Exploiting the circulant structure of tracking-by-detection with kernels”, dans *ECCV*, 2012.

——, “High-speed tracking with kernelized correlation filters”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.

Z. Hong, C. Wang, X. Mei, D. Prokhorov, et D. Tao, “Tracking using multilevel quantizations”, dans *ECCV*, 2014.

M. Isard et A. Blake, “Condensation - Conditional Density Propagation for Visual Tracking”, *International Journal of Computer Vision*, vol. 29, pp. 5–28, 1998.

X. Jia, H. Lu, et M.-H. Yang, “Visual tracking via adaptive structural local sparse appearance model”, dans *CVPR*, 2012, pp. 1822–1829.

N. Y. Khan, B. McCane, et G. Wyvill, “Sift and surf performance evaluation against various image deformations on benchmark dataset”, dans *International Conference on Digital Image Computing : Techniques and Applications*, 2011, pp. 501–506.

M. Kristan, R. Pflugfelder, et G. Nebehay, “The visual object tracking VOT2014 challenge results”, dans *ECCV Workshop*, 2014, pp. 1–27.

——, “The visual object tracking VOT2015 challenge results”, dans *ICCV Workshop*, 2015.

A. Krizhevsky, I. Sutskever, et G. Hinton, “Imagenet classification with deep convolutional neural networks”, dans *NIPS*, 2012.

H. Kuhn, “The Hungarian Method for the assignment problem”, *Naval Research Logistics Quarterly*, vol. 2, pp. 83–97, 1955.

J. Kwon et K. M. Lee, “Visual tracking decomposition”, dans *CVPR*, 2010.

J. Lasserre, C. Bishop, et T. Minka, “Principled Hybrids of Generative and Discriminative Models”, dans *CVPR*, 2006.

S. Leutenegger, M. Chli, et R. Siegwart, “BRISK : Binary Robust Invariant Scalable Keypoints”, dans *ICCV*, 2011, pp. 2548–2555.

Y. Li, J. Zhu, et S. C. Hoi, “Reliable Patch Trackers : Robust Visual Tracking by Exploiting Reliable Patches”, dans *CVPR*, 2015, pp. 353–361.

Y. Liu, W. Zhou, H. Yin, et N. Yu, “Tracking Based on SURF and Superpixel”, dans *Sixth International Conference on Image and Graphics*, 2011, pp. 714–719.

J. Long, E. Shelhamer, et T. Darrell, “Fully convolutional networks for semantic segmentation”, dans *CVPR*, 2015.

D. G. Lowe, “Distinctive image features from scale-invariant keypoints”, *IJCV*, vol. 60, no. 2, pp. 91–110, 2004.

C. Ma, J. Huang, X. Yang, et M. Yang, “Hierarchical Convolutional Features for Visual Tracking”, dans *ICCV*, 2015.

L. Ma, X. Zhang, W. Hu, J. Xing, J. Lu, et J. Zhou, “Local Subspace Collaborative Tracking”, dans *ICCV*, 2015.

X. Mei et H. Ling, “Robust visual tracking using l1 minimization”, dans *ICCV*, 2009.

K. Mikolajczyk et C. Schmid, “A performance evaluation of local descriptors”, *TPAMI*, vol. 27, no. 10, 2005.

H. Nam et B. Han, “Learning multi-domain convolutional neural networks for visual tracking”, dans *CoRR*, 2015.

G. Nebehay et R. Pflugfelder, “Consensus-based matching and tracking of keypoints for object tracking”, dans *WACV*, 2014.

——, “Clustering of Static-Adaptive Correspondences for Deformable Object Tracking”, dans *CVPR*, 2015.

Y. Ng et M. Jordan, “A comparison of logistic regression and naive Bayes”, *MIT Press*, vol. 14, pp. 841–848, 2001.

H. Possegger, T. Mauthner, et H. Bischof, “In defense of color-based model-free tracking”, dans *CVPR*, 2015.

X. Ren et J. Malik, “Tracking as repeated figure/ground segmentation”, dans *CVPR*, 2007, pp. 1–8.

D. Ross, J. Lim, R. Lin, et M. Yang, “Incremental learning for robust visual tracking”, *International Journal of Computer Vision*, vol. 77, no. 1, pp. 125–141, 2008.

C. Rother, V. Kolmogorov, et A. Blake, “Grabcut : Interactive foreground extraction using iterated graph cuts”, *ACM Transactions on Graphics (TOG)*, vol. 23, no. 3, pp. 309–314, 2004.

B. Schölkopf et A. Smola, “Learning with kernels : Support vector machines, regularization, optimization, and beyond”, *MIT Press*, 2002.

J. Shi et J. Malik, “Motion segmentation and tracking using normalized cuts”, dans *ICCV*, 2006, pp. 1154–1160.

Y. Sui, Y. Tang, et L. Zhang, “Discriminative Low-Rank Tracking”, dans *ICCV*, 2015.

T. Tuytelaars et K. Mikolajczyk, “Local invariant feature detectors : a survey”, *FTCGV*, vol. 3, 2008.

V. Vapnik, *The Nature of Statistical Learning Theory.* New York : Springer, 1995.

J. Wang et Y. Yagi, “Many-to-Many Superpixel Matching for Robust Tracking”, *IEEE Transaction on Cybernetics*, vol. 44, no. 7, pp. 1237–1248, 2014.

N. Wang, J. Shi, D.-Y. Yeung, et J. Jia, “Understanding and Diagnosing Visual Tracking Systems”, dans *ICCV*, 2015.

S. Wang, H. Lu, F. Yang, et M.-H. Yang, “Robust Superpixel Tracking”, *IEEE Trans. Image Process.*, vol. 23, no. 4, pp. 1639–1651, 2014.

W. Wang et R. Nevatia, “Multistore tracker (muster) : A cognitive psychology inspired approach to object tracking”, dans *CVPR*, 2015, pp. 749–758.

L. Wen, D. Du, Z. L, S. Li, et M. Yang, “Joint Online Tracking and Segmentation”, dans *CVPR*, 2015.

Y. Wu, J. Lim, et M.-H. Yang, “Online Object Tracking : A Benchmark”, dans *CVPR*, 2013.

F. Yang, H. Lu, et M.-H. Yang, “Learning structured visual dictionary for object tracking”, *Image and Vision Computing*, vol. 31, no. 12, pp. 992–999, 2013.

W. Zhong, H. Lu, et M.-H. Yang, “Robust object tracking via sparsity-based collaborative model”, dans *CVPR*, 2012, pp. 1838–1845.

Q. Zhou, H. Lu, et M.-H. Yang, “Online multiple support instance tracking”, dans *Proc. IEEE Conf. Automatic Face and Gesture Recognition*, 2011, pp. 545–552.